برمجة التحكم المنطقى *P.L.C.*

الجزء الثانى

إعداد

ريمون كمال

معهد السالزيان الإيطالى "**دون بوسكو**"

٢شارع عبد القادر طه – الساحل   ت: ٢٤٥٧٩٦٥٠ – ٢٤٥٧٦٧٩٤

معهد فنى – معهد صناعى

دورات تدريبية سريعة مركزة

دورات تدريبية تعليمية للمدرسين

المراجع

١. أجهزة التحكم المبرمج وتطبيقاتها العملي.

1. Controllore a logica programmabile P. Bani.
2. Siemens Programmable Controller Manual.

طبعة جديدة

2012

أسم الكتاب: **برمجة التحكم المنطقى *P.L.C.***

الجزء الثانى

طباعة

رقم الإيداع:

الترقيم الدولى:

شكر و إهداء

أهدى هذا الكتاب إلى أبي وأمي الذين لهم كل الفضل بأن أعمل في هذا المجال وهم الذين شجعوني على عمل هذا الكتاب بكل جهد وإخلاص شاكر الله و إياهم وكل من ساهم في تقديمه.

وأشكر أيضاً كل المعلمين الأفاضل الذين ساعدوا على خروج هذا الكتاب إلى المليء.

- **المدير الإيطالي للمعهد:** الأب رينسو ليوناردوسى
- **الناظر السابق للمعهد:** الأب بيرناردو أشيربونى
- **مدير الدورات التدريبية:** أ. ماجد چورچ
- **أستاذ التحكم:** أ. نبيل رزق **–** أ. وجية جرجس
- **أستاذ التكيف والتبريد:** أ. إميل فتح الله
- **أستاذ الـــ PLC:** أ. ماجد موريس **–** أ. ماجد عريان **–** أ. چيوليو جالو **–** أ. محسن أنطون

أشكر كل من أرسلوا لى التعليقات بخصوص الجزء الأول من الكتاب وقد حاولت قدر المستطاع تلبية متطلباتهم فى الجزء الثانى وأتشرف باستقبال المزيد من تعليقات السادة القراء بخصوص هذا لجزء من الكتاب على عنوان البريد التالى

plcbook@hotmail.com

مقدمة

نظراً للتقدم العلمى السريع المرتبط بالمجال الصناعى وخاصة من الناحية الكهربية أصبح لا غنى عن الربط بين عالم الصناعة وبين التكنولوچيا العصرية ويتمثل هذا الربط بواسطة استخدام أجهزة التحكم المنطقى بمختلف أنواعها والتى تستحق أن تسمى بالأجهزة الذكية نظراً لما تقدمه فى المجال الصناعى من: سهولة فى تصميم البرامج، ومرونة فى أكتشاف الأعطال، ومساعدة فى حل المشاكل، ...الخ

و نظراً لصعوبة ترجمة بعض المصطلحات الخاصة بهذا المجال وخاصة لكى لا تفقد المعنى التقنى أو الفنى لها، تمت كتابتها بلغتها الأصلية لذلك لا تهتم كثيراً عزيزى القارئ بهذه المصطلحات فستكون بسيطة ومفهومة بمجرد ما أن تتعمق بفهم فى هذا المجال.

هكذا أيضاً لا تتعجل عزيزى القارئ فى النظر إلى مواضيع متباعدة خاصة أن كنت بمبتدئ فى هذا المجال وهذا لأنه قد تم شرح المنهج بطريقة متسلسلة ولذلك يفضل للقارئ قراءة المواضيع بالتسلسل التى كتبت به لفهم جميع الأمور دون تخبط.

و خاصاً لفهم التمارين لا يشترط فقط القراءة بترتيب بل يجب أيضاً أن تربط كل شرح و كل رمز بالرسم الموجود ولا تقوم بالقراءة بطريقة عابرة.

تم شرح البرمجة بطريقة عامة دون اللجؤ إلى ماركة بعينها وهذا لكى يخدم كل من يعمل مع وحدات التحكم المنطقى بمختلف أنواعها.

تنقسم معرفة أجهزة التحكم المنطقى إلى أمور عديدة من أهمها:

تصميم برامج – اكتشاف أعطال – حل مشاكل

قد تم التركيز بشكل كبير فى الجزء الأول من هذا الكتاب على معرفة أساسات تصميم البرامج بطريقة سلسة وباستخدام أسهل لغات البرمجة ولذلك أقدم لكم الجزء الثانى لاستكمال شرح تصميم البرامج باستخدام أساليب ذات أكثر تطور ويتم الآن تحضير جزء خاص "بأكتشاف الأعطال – بحل المشاكل وبتصميم التمارين والأفكار العملية".

لذلك أقدم لكم الجزء الثانى من هذا الكتاب لخدمة كل من يدرس أو يعمل فى هذا المجال و أتمنى من الله أن يجد كل من يقرأ هذا الكتاب نفعاً له.

المـــؤلـــف

الباب الأول

العمليات الحسابية للأرقام الصحيحة و العشرية

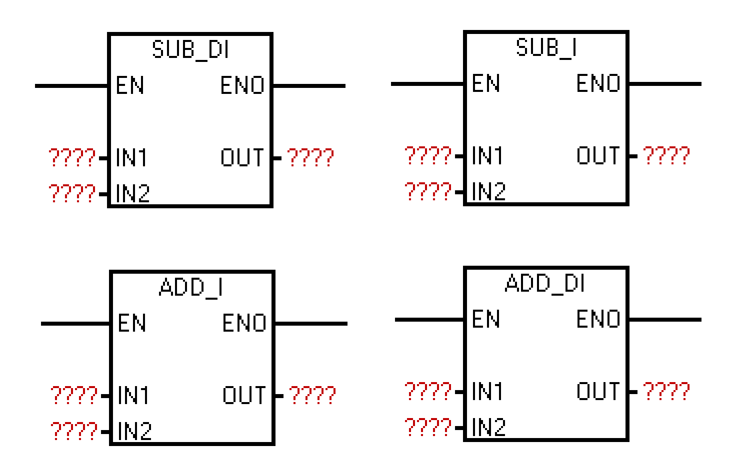
- أنواع العمـــــــــــــــــــــليات الحسابية.
- العمليـــــــــــــات الحسابية للأرقام الصحيحة.
- العمليـــــــــــــات الحسابية للأرقام العشرية.
- الفـــــــــــــــــــــــــرق بين النوعين.
- طريقة توصيل العمليـــات الحسابيـــــــــــة.
- مـــلاحظات هامة بخصوص العمليات الحسابية.
- تمارين تطبيقية على العمليات الحسابيـــــــة.

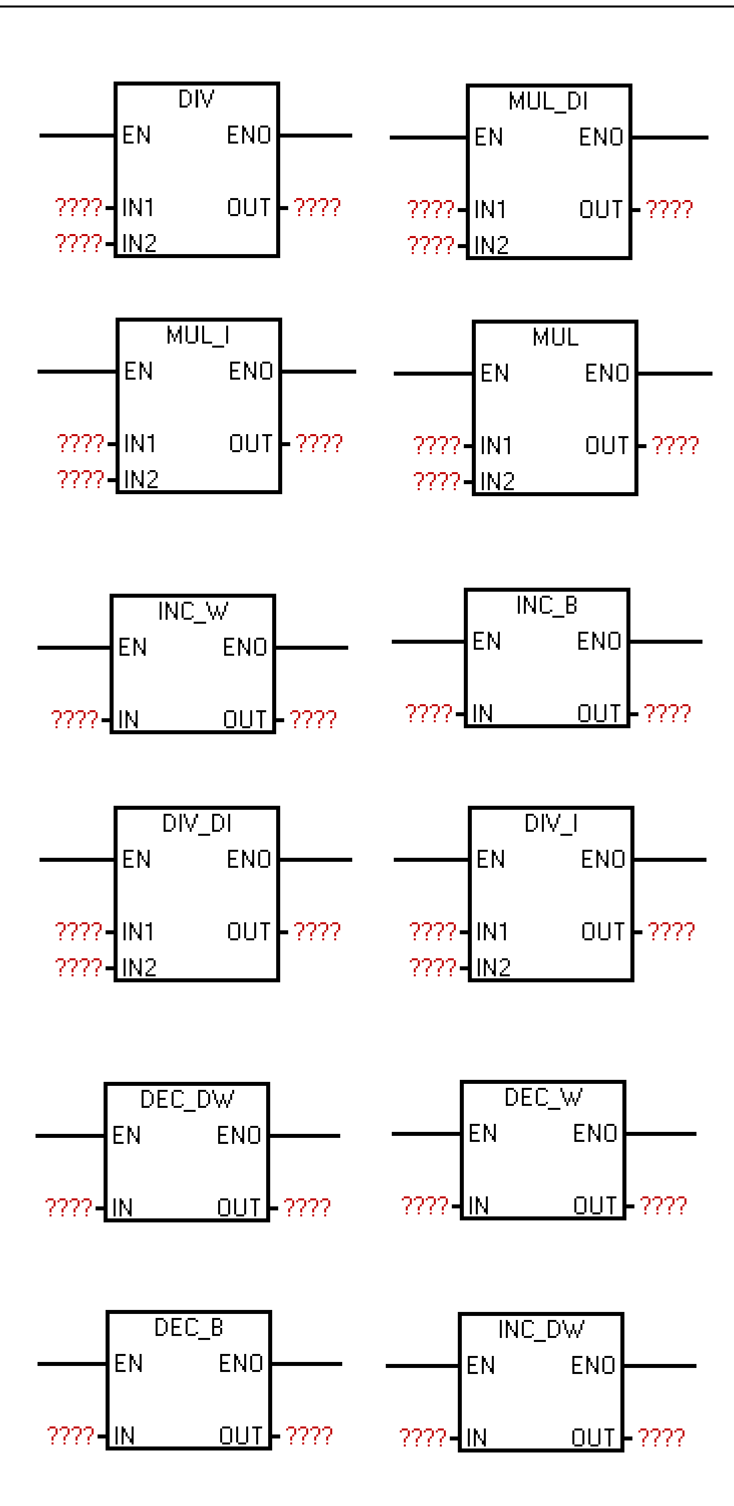
العمليات الحسابية:

تستخدم العمليات الحسابية فى بعض البرامج التى تحتوى على متغيرات أو التى يتم فيها التحويل من أى وحدة قياس إلى أى وحدة قياس مختلفة أو حتى للقيام بمعدلات من الدرجة الأولى, أو الثانية, أو ......

تنقسم العمليات الحسابية إلى نوعين, النوع الأول هو العمليات الحسابية للأرقام الصحيحة و الثانية هى العمليات الحسابية للأرقام العشرية.

العمليات الحسابية للأرقام الصحيحة:




DIV
EN
ENO
IN1
OUT
IN2
????
MUL_DI
EN
ENO
IN1
OUT
IN2
MUL_I
EN
ENO
IN1
OUT
IN2
MUL
EN
ENO
IN1
OUT
IN2
INC_W
EN
ENO
IN
OUT
INC_B
EN
ENO
IN
OUT
DIV_DI
EN
ENO
IN1
OUT
IN2
DIV_I
EN
ENO
IN1
OUT
IN2
DEC_DW
EN
ENO
IN
OUT
DEC_W
EN
ENO
IN
OUT
DEC_B
EN
ENO
IN
OUT
INC_DW
EN
ENO
IN
OUT

كل العمليات التى تتم بواسطة **العمليات الحسابية للأرقام الصحيحة** يجب أن تحتوى فقط على أرقام صحيحة و يجب أن تكون النواتج هى أيضاً صحيحة فمثلاً:

- **لا يمكن استخدام العمليات الحسابية الصحيحة لجمع أرقام غير صحيحة فمثلاً:**

  لا يمكن جمع رقم بقيمة ٢,٥ و قيمة ٦,١ لأنها ليست أرقام صحيحة.

- **لا يمكن استخدام العمليات الحسابية الصحيحة لطرح أرقام غير صحيحة فمثلاً:**

  لا يمكن طرح رقم بقيمة ٨,٥ من رقم ٦,١ لأنها ليست أرقام صحيحة.

- **لا يمكن استخدام العمليات الحسابية الصحيحة لقسمة أرقام غير صحيحة فمثلاً:**

  لا يمكن قسمة رقم بقيمة ٠,١ و قيمة ٥,٣ لأنها ليست أرقام صحيحة.

- **لا يمكن استخدام العمليات الحسابية الصحيحة لضرب أرقام غير صحيحة فمثلاً:**

  لا يمكن ضرب رقم بقيمة ٨,١٠ و قيمة ٣,٢ لأنها ليست أرقام صحيحة.

ملاحظة:

يجب أيضاً مراعاة الخرج بالنسبة للمعادلة أى مراعاة أن الناتج يجب أن يكون رقم صحيح و أن يكون الناتج موضوع على ذاكرة بالحجم الصحيح, فمثلاً:

- لا يمكن استخدام العمليات الحسابية الصحيحة بحجم byte لضرب قيمة ٢٠٠ و قيمة ٢٠٠ باستخدام MUL-B لأن الناتج لا يمكن أن يكتب على byte.

  لتجنب هذه المشكلة يتم التعامل مع عملية حسابية أخرى ذات حجم أكبر مثل: word.

- لا يمكن استخدام العمليات الحسابية الصحيحة لقسمة أرقام صحيحة مثل قيمة ١٠ و قيمة ٣ لأن الناتج لن سيكون رقم صحيح.
- لتجنب هذه المشكلة يتم التعامل مع عملية حسابية أخرى غير صحيحة كما سنوضح بعد قليل.

م	الاسم	الشرح	الشكل
١	ADD_I عملية جمع أرقام صحيحة بحجم Word.	عمليات جمع بحجم Word تقوم بجمع أرقام صحيحة (IN1 و IN2) ويكتب الناتج (OUT) على ذاكرة Word.	ADD_I EN ENO ????-IN1 OUT-???? ????-IN2
٢	ADD_DI عملية جمع أرقام صحيحة بحجم Dword.	عمليات جمع بحجم Dword تقوم بجمع أرقام صحيحة (IN1 و IN2) ويكتب الناتج (OUT) على ذاكرة Dword.	ADD_DI EN ENO ????-IN1 OUT-???? ????-IN2
٣	SUB_I عملية طرح أرقام صحيحة بحجم Word.	عمليات طرح بحجم Word تقوم بطرح أرقام صحيحة (IN2 من IN1) ويكتب الناتج OUT على ذاكرة Word.	SUB_I EN ENO ????-IN1 OUT-???? ????-IN2
٤	SUB_DI عملية طرح أرقام صحيحة بحجم Dword.	عمليات طرح بحجم Dword تقوم بطرح أرقام صحيحة (IN2 من IN1) ويكتب الناتج OUT على ذاكرة Dword.	SUB_DI EN ENO ????-IN1 OUT-???? ????-IN2

٥	MUL عملية ضرب أرقام صحيحة بحجم Dword/Word.	عمليات ضرب بحجم Word للدخل و Dword للخرج، تقوم بضرب أرقام صحيحة بحجم Word (IN1 و IN2) ويكتب الناتج OUT على ذاكرة Dword.	MUL EN ENO ????-IN1 OUT-???? ????-IN2
٦	MUL_I عملية ضرب أرقام صحيحة بحجم Word.	عمليات ضرب بحجم Word تقوم بضرب أرقام صحيحة (IN1 و IN2) ويكتب الناتج OUT على ذاكرة Word.	MUL_I EN ENO ????-IN1 OUT-???? ????-IN2
٧	MUL_DI عملية ضرب أرقام صحيحة بحجم Dword.	عمليات ضرب بحجم Dword تقوم بضرب أرقام صحيحة (IN1 و IN2) ويكتب الناتج OUT على ذاكرة Dword.	MUL_DI EN ENO ????-IN1 OUT-???? ????-IN2
٨	DIV عملية قسمة أرقام صحيحة بحجم Dword/Word.	عمليات قسمة بحجم Word للدخل و Dword للخرج، تقوم بقسمة أرقام صحيحة بحجم Word (IN1 على IN2) ويكتب الناتج OUT على ذاكرة Dword.	DIV EN ENO ????-IN1 OUT-???? ????-IN2

٩	DIV_I عملية قسمة أرقام صحيحة بحجم Word.	عمليات قسمة بحجم Word تقوم بقسمة أرقام صحيحة (IN1 على IN2) ويكتب الناتج OUT على ذاكرة Word.	DIV_I EN ENO ???? IN1 OUT ???? ???? IN2
١٠	DIV_DI عملية قسمة أرقام صحيحة بحجم Dword.	عمليات قسمة بحجم Word تقوم بقسمة أرقام صحيحة (IN1 على IN2) ويكتب الناتج OUT على ذاكرة Word.	DIV_DI EN ENO ???? IN1 OUT ???? ???? IN2
١١	INC_B عملية إضافة تصاعدية بحجم Byte.	عمليات الإضافة تصاعدياً بحجم Byte بحيث يضاف واحد على الدخل IN و ينقل إلى الخرج OUT على ذاكرة بحجم Byte.	INC_B EN ENO ???? IN OUT ????
١٢	INC_W عملية إضافة تصاعدية بحجم Word.	عمليات ألإضافة تصاعدياً بحجم Word بحيث يضاف واحد على الدخل IN و ينقل إلى الخرج OUT على ذاكرة بحجم Word.	INC_W EN ENO ???? IN OUT ????
١٣	INC_DW عملية إضافة تصاعدية بحجم Dword.	عمليات ألإضافة تصاعدياً بحجم Dword بحيث يضاف واحد على الدخل IN و ينقل إلى الخرج OUT على ذاكرة بحجم Dword.	INC_DW EN ENO ???? IN OUT ????

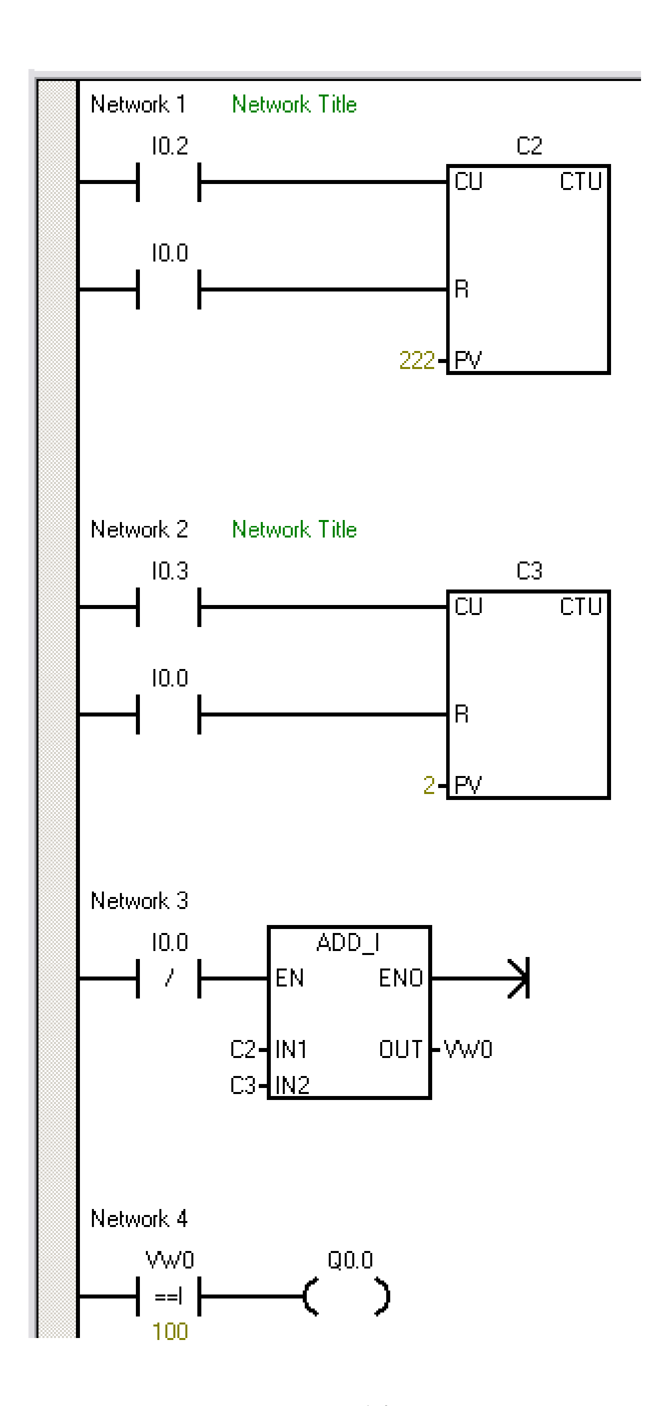
١٤	DEC_B عملية طرح تنازلية بحجم Byte.	عمليات الطرح تنازلياً بحجم Byte بحيث يطرح واحد من الدخل IN و ينقل إلى الخرج OUT على ذاكرة بحجم Byte.	DEC_B EN ENO ????-IN OUT-????
١٥	DEC_W عملية طرح تنازلية بحجم Word.	عمليات الطرح تنازلياً بحجم Word بحيث يطرح واحد من الدخل IN و ينقل إلى الخرج OUT على ذاكرة بحجم Word.	DEC_W EN ENO ????-IN OUT-????
١٦	DEC_DW عملية طرح تنازلية بحجم Dword.	عمليات الطرح تنازلياً بحجم Dword بحيث يطرح واحد من الدخل IN و ينقل إلى الخرج OUT على ذاكرة بحجم Dword.	DEC_DW EN ENO ????-IN OUT-????

أمثلة (تمارين عملية):

١– قم بتنفيذ دائرة تحكم منطقية لمصنع يحتوى على خطين أنتاج بحيث تضاء اللمبة عندما يكون حاصل مجموع القطع المنتجة من الخطين هو ١٠٠ قطعة.

عدد الدخل	نوع الدخل	أسم الدخل
١	n.o.	I0.0/S1
٢	n.o.	I0.2/S2
٣	n.o.	I0.3/S3
عدد العدادات	**نوع العدادات**	**أسم العدادات**
١	CTU	C2
٢	CTU	C3
عدد مفاتيح المقارنة	**نوع مفاتيح المقارنة**	**أسم مفاتيح المقارنة**
١	==I	VW0
عدد العمليات	**نوع العمليات**	**أسم العمليات**
١	ADD_I	ADD_I
عدد الخرج	**نوع الخرج**	**أسم الخرج**
١	لمبة	Q0.0

البرنامج:



Network 1
Network Title
I0.2
C2
CU
CTU
I0.0
R
222
PV
Network 2
Network Title
I0.3
C3
CU
CTU
I0.0
R
2
PV
Network 3
I0.0
ADD_I
EN
ENO
C2
IN1
OUT
VW0
C3
IN2
Network 4
VW0
==I
100
Q0.0

الشرح:

Network1:

عند مرور أى قطعة على خط الإنتاج الأول أمام الحساس I0.2 فأنة يرسل أشارة إلى العداد C2 بشرط أن يكون المفتاح I0.0 مفتوح.

Network2:

عند مرور أى قطعة على خط الإنتاج الثانى أمام الحساس I0.3 فأنة يرسل أشارة إلى العداد C3 بشرط أن يكون المفتاح I0.0 مفتوح.

Network3:

يقوم بجمع عدد القطع التى تم عدها سواء بواسطة العداد الخاص بخط الإنتاج الأول C2 أو بواسطة العداد الخاص بخط الإنتاج الثانى C3 وكتابة المجموع فى الذاكرة VW0.

Network4:

عندما تصبح قيمة الذاكرة VW0 تساوى ١٠٠ يصبح مفتاح المقارنة مغلق فتضاء اللمبة.

٢- قم بتنفيذ دائرة تحكم منطقية لمصنع يحتوى على خط أنتاج بحيث تمر الكرتونة أمام الحساس مع مراعاة أن كل كرتونة تحتوى على ١٢ زجاجة, صمم برنامج لمعرفة عدد الزجاجات و ليس الكراتين وتضاء لمبة أشارة عندما يصل عدد الزجاجات إلى ١٢٠.

عدد الدخل	نوع الدخل	أسم الدخل
١	n.o.	I0.0/S1
٢	n.o.	I0.1/S2
عدد العدادات	**نوع العدادات**	**أسم العدادات**
١	CTU	C0
عدد مفاتيح المقارنة	**نوع مفاتيح المقارنة**	**أسم مفاتيح المقارنة**
١	==I	VW24
عدد العمليات	**نوع العمليات**	**أسم العمليات**
١	MUL_I	MUL_I
عدد الخرج	**نوع الخرج**	**أسم الخرج**
١	لمبة	Q1.1/K1M

الشرح:

:Network1

عند مرور أى كرتونة على خط الإنتاج أمام الحساس I0.1 فأنة يرسل أشارة إلى العداد C0 بشرط أن يكون المفتاح I0.0 مفتوح.

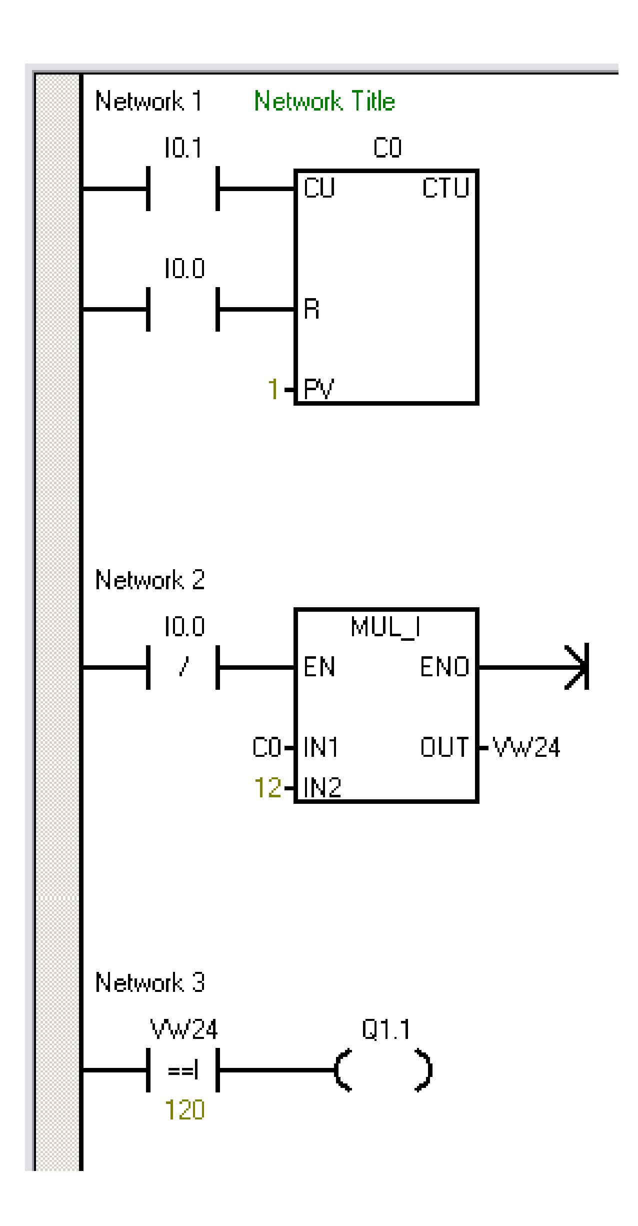
:Network2

يقوم البرنامج بضرب عدد الكراتين التى تم عدها بواسطة العداد C0 فى عدد الزجاجات ثم كتابة المجموع فى الذاكرة VW24.

:Network3

عندما تصبح قيمة الذاكرة VW24 تساوى ١٢٠ يصبح مفتاح المقارنة مغلق فتضاء اللمبة.

البرنامج:


Network 1
Network Title
I0.1
C0
CU
CTU
I0.0
R
1
PV
Network 2
I0.0
MUL_I
EN
ENO
/
C0
IN1
OUT
VW24
12
IN2
Network 3
VW24
Q1.1
==I
120

٣- قم بتنفيذ دائرة تحكم منطقية لمعادلة تقوم بتحويل درجة الحرارة من **Kelvin** إلى **Celsius** بحيث أذا كانت قيمة درجة الحرارة تحت الصفر تضيء لمبة حمراء.

علماً بأن المعادلة الخاصة بالتحويل هى. Kelvin=Celsius + 273

عدد الدخل	نوع الدخل	أسم الدخل
١	n.c.	I0.0/Stop
٢	n.o.	I0.1/Start
عدد المتغيرات	**نوع المتغيرات**	**أسم المتغيرات**
١	word	VW0(celsius)
٢	word	VW2(273)
٣	word	VW4(kelvin)
عدد العمليات	**نوع العمليات**	**أسم العمليات**
١	ADD_I	ADD_I
عدد الخرج	**نوع الخرج**	**أسم الخرج**
١	لمبة	Q1.1/K1M

الشرح:

:Network1

بالضغط على I0.1 وبشرط أن يكون I0.0 مغلق فيعمل الريليه M0.7

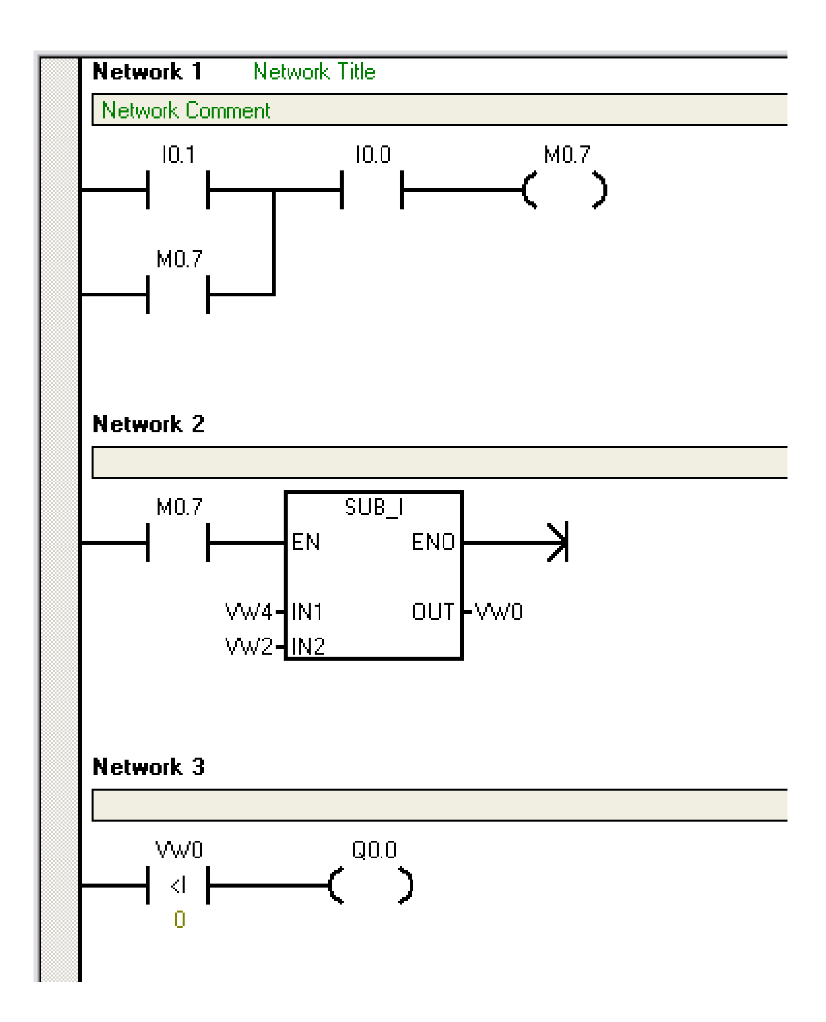
:Network2

يقوم بطرح قيمة المتغير VW2 الذى يمثل درجة الحرارة بالـ Kelvin من قيمة المتغير الآخرة VW2 التى تمثل الفرق بين القيمتين "273".

:Network3

عندما تصبح قيمة الـ VW0 أقل من صفر سوف تضاء لمبة لتشير أن درجة الحرارة بالسالب.

البرنامج:


Network 1
Network Title
Network Comment
I0.1
I0.0
M0.7
M0.7
Network 2
M0.7
SUB_I
EN
ENO
VW4
IN1
OUT
VW0
VW2
IN2
Network 3
VW0
<I
0
Q0.0

٤– قم بتنفيذ دائرة تحكم منطقية لمعادلة تقوم بنفس عمل العداد حيث يعمل الخرج عندما يصل العدد إلى ٣٠٠٠٠٠ عدة.

علماً بأن المعادلة الخاصة بالعداد. VD0 = VD0 + 1

عدد الدخل	نوع الدخل	أسم الدخل
١	n.o.	I0.0/Stop
٢	n.o.	I0.1/Start
عدد المتغيرات	**نوع المتغيرات**	**أسم المتغيرات**
١	D.word	VD0
عدد العمليات	**نوع العمليات**	**أسم العمليات**
١	INC_DW	INC_DW
عدد الخرج	**نوع الخرج**	**أسم الخرج**
١	لمبة	Q1.0/lamp

الشرح:

:Network1

بالضغط على I0.1 سوف يتم إضافة واحد إلى المتغير VD0 فيعمل تماماً مثل العداد وقد تم استخدام مفتاح الـ positive edge لكى تكون كل ضغطة على المفتاح تعادل عدة واحدة فقط وليس أكثر.

:Network2

عندما تصبح قيمة المتغير VD0 تساوى 300000 فسوف يعمل الخرج Q1.0 تلقائياً.

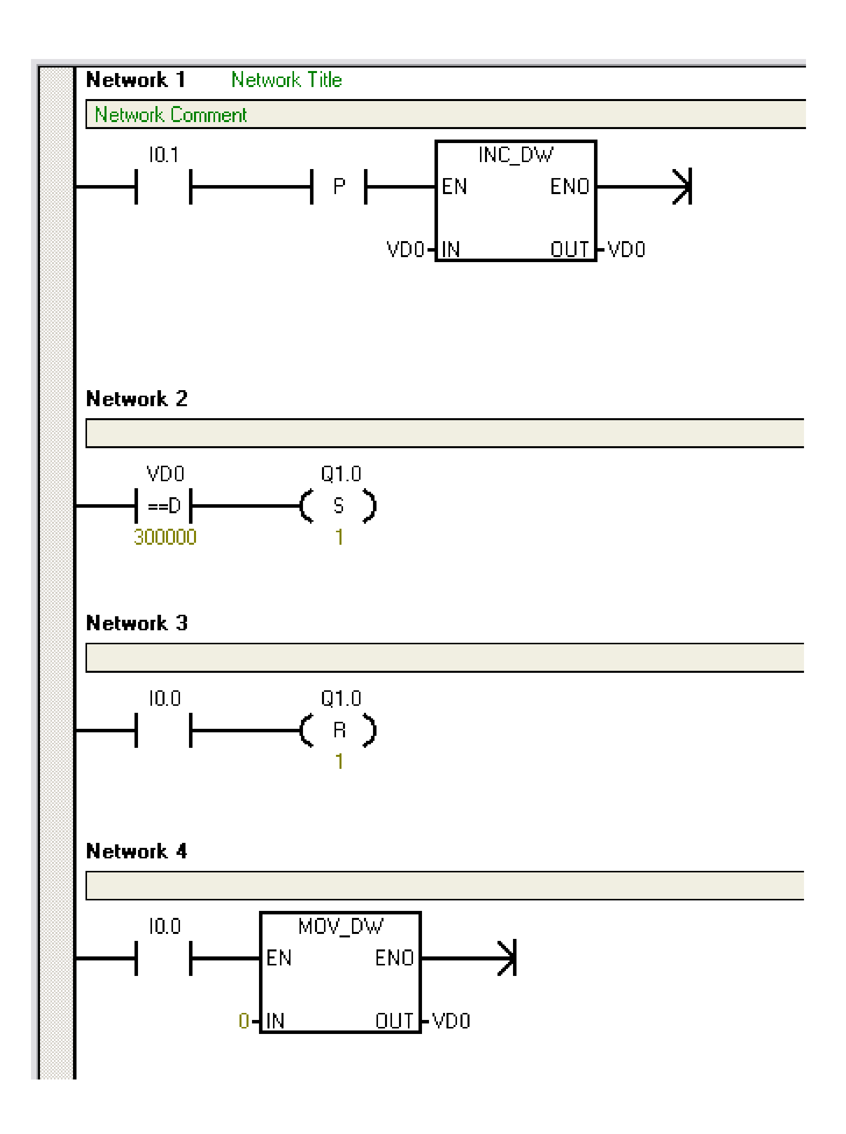
:Network3

بالضغط على I0.0 سوف يتم عمل reset أى إيقاف للخرج Q1.0.

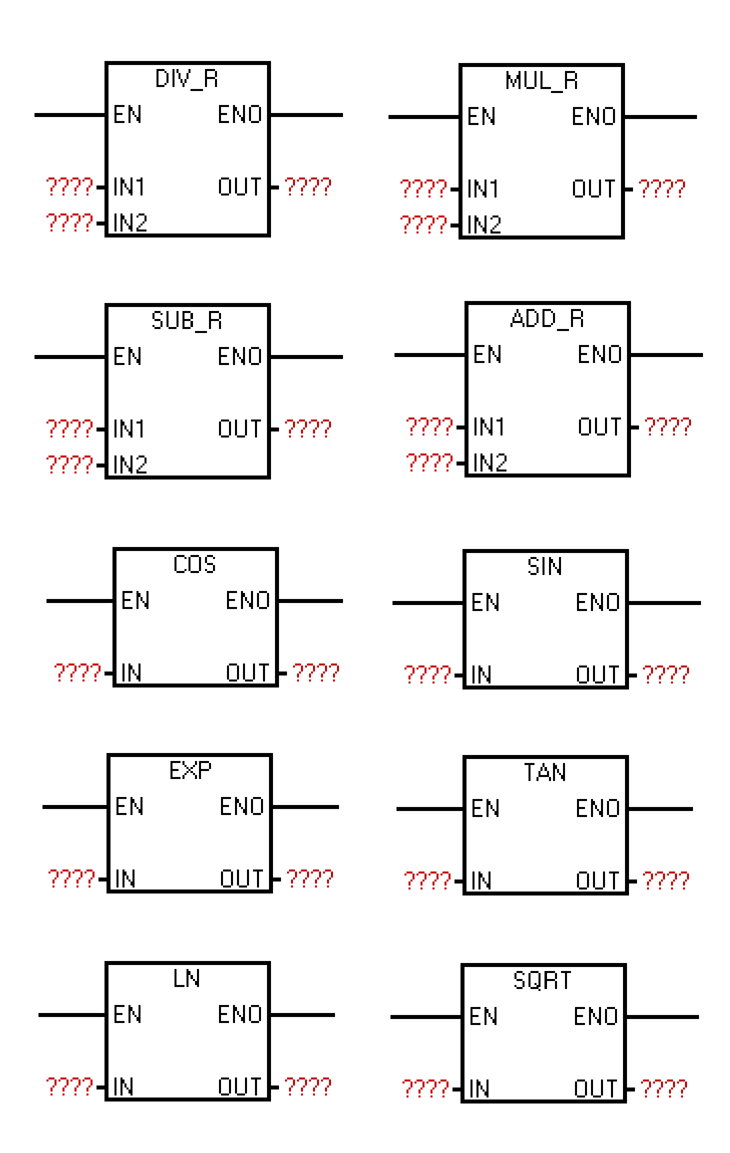
:Network4

بالضغط على I0.0 أيضاً سوف يتم نقل قيمة صفر إلى المتغير VD0 لكى يتمكن العامل من البدء من جديد بدايتاً من الصفر كما كان الوضع فى البداية.

البرنامج.


Network 1
Network Title
Network Comment
I0.1
P
INC_DW
EN
ENO
VD0
IN
OUT
VD0
Network 2
VD0
==D
300000
Q1.0
S
1
Network 3
I0.0
Q1.0
R
1
Network 4
I0.0
MOV_DW
EN
ENO
0
IN
OUT
VD0

العمليات الحسابية للأرقام العشرية:



كل العمليات التى تتم بواسطة العمليات الحسابية للأرقام العشرية يجب أن تحتوى فقط على أرقام عشرية و يجب أن تكون النواتج هى أيضاً عشرية فمثلاً:

- **لا يمكن استخدام العمليات الحسابية العشرية لجمع أرقام غير عشرية فمثلاً:**

لا يمكن جمع رقم بقيمة ٢ و قيمة ٦ لأنها ليست أرقام عشرية.

- **لا يمكن استخدام العمليات الحسابية العشرية لطرح أرقام غير عشرية فمثلاً:**

لا يمكن طرح رقم بقيمة ٨ من رقم ١ لأنها ليست أرقام عشرية.

- **لا يمكن استخدام العمليات الحسابية العشرية لقسمة أرقام غير عشرية فمثلاً:**

لا يمكن قسمة رقم بقيمة ١ و قيمة ٣ لأنها ليست أرقام عشرية.

- **لا يمكن استخدام العمليات الحسابية العشرية لضرب أرقام غير عشرية فمثلاً:**

لا يمكن ضرب رقم بقيمة ١٠٨ و قيمة ٢٣ لأنها ليست أرقام عشرية.

ملاحظة:

لا توجد مشكلة بالنسبة للمعادلة أى لا داعى لمراعاة الناتج من الناحية الخاصة بالذاكرة لأنه:

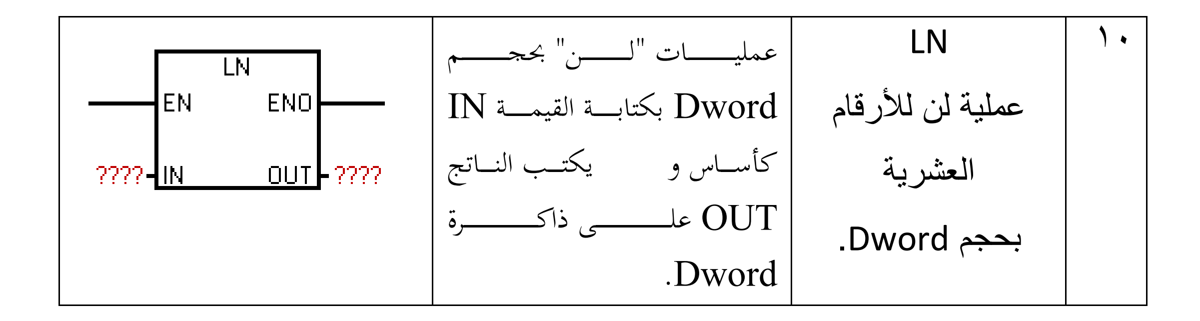
- يتم استخدام ذاكرة بحجم Dword مع العمليات الحسابية العشرية وهو أكبر حجم للذاكرة فى الـ PLC كما وضح فى الجزء الأول من هذا الكتاب.

لا توجد مشكلة بالنسبة للمعادلة أى لا داعى لمراعاة الناتج من الناحية الخاصة بنوع الناتج لأنه:

- فى حالة أن كان الناتج هو رقم صحيح بالصدفة مثلاً فلا توجد أى مشكلة لأنه يتم إضافة "٠," إلى الرقم فلا تتغير القيمة ولكن يصبح الرقم عشرى تلقائياً.

م	الاسم	الشرح	الشكل
١	ADD_R عملية جمع أرقام عشرية بحجم Dword.	عمليات جمع بحجم Dword تقوم بجمع أرقام عشرية (IN1 و IN2) ويكتب الناتج (OUT) على ذاكرة Dword.	ADD_R EN ENO ????-IN1 OUT-???? ????-IN2
٢	SUB_R عملية طرح أرقام عشرية بحجم Dword.	عمليات طرح بحجم Dword تقوم بطرح أرقام عشرية (IN1 و IN2) ويكتب الناتج (OUT) على ذاكرة Dword.	SUB_R EN ENO ????-IN1 OUT-???? ????-IN2
٣	MUL_R عملية ضرب أرقام عشرية بحجم Dword.	عمليات ضرب بحجم Dword تقوم بضرب أرقام عشرية (IN1 و IN2) ويكتب الناتج (OUT) على ذاكرة Dword.	MUL_R EN ENO ????-IN1 OUT-???? ????-IN2
٤	DIV_R عملية ضرب أرقام عشرية بحجم Dword.	عمليات قسمة بحجم Dword تقوم بقسمة أرقام عشرية (IN1 و IN2) ويكتب الناتج (OUT) على ذاكرة Dword.	DIV_R EN ENO ????-IN1 OUT-???? ????-IN2

٥	SQRT عملية جزر تربيعي أرقام عشرية بحجم Dword.	عمليات الجزر التربيعي بحجم Dword للدخل IN ويكتب الناتج OUT على ذاكرة Dword.	SQRT EN ENO ????-IN OUT-????
٦	COS عملية جتا للأرقام العشرية بحجم Dword.	عمليات "جتا" بحجم Dword للدخل IN ويكتب الناتج OUT على ذاكرة Dword.	COS EN ENO ????-IN OUT-????
٧	SIN عملية جا للأرقام العشرية بحجم Dword.	عمليات "جا" بحجم Dword للدخل IN ويكتب الناتج OUT على ذاكرة Dword.	SIN EN ENO ????-IN OUT-????
٨	TAN عملية ظا للأرقام العشرية بحجم Dword.	عمليات "ظا" بحجم Dword للدخل IN ويكتب الناتج OUT على ذاكرة Dword.	TAN EN ENO ????-IN OUT-????
٩	EXP عملية الأس للأرقام العشرية بحجم Dword.	عمليات "الأس" باستخدام قيمة ١٠ كأساس بحيث يكتب رقم الأس في الدخل IN ويكتب الناتج OUT على ذاكرة Dword.	EXP EN ENO ????-IN OUT-????



١٠	LN عملية لن للأرقام العشرية بحجم Dword.	عمليات "لن" بحجم Dword بكتابة القيمة IN كأساس و يكتب الناتج OUT على ذاكرة Dword.	LN EN ENO ???? IN OUT ????

أمثلة (تمارين عملية):

١ – قم بتنفيذ دائرة تحكم منطقية لقانون فيثاغورث

علماً بأن المعادلة الخاصة بقانون فيثاغورث هي

$$C = \sqrt{a^2 + b^2}$$

عدد الدخل	نوع الدخل	أسم الدخل
١	n.c.	I0.0/Stop
٢	n.o.	I0.1/Start
عدد المتغيرات	**نوع المتغيرات**	**أسم المتغيرات**
١	D.word	VD0(a)
٢	D.word	VD2(b)
٣	D.word	VD4(C)
٤	D.word	VD20(a^2)
٥	D.word	VD22(b^2)
٦	D.word	VD40($\sqrt{a^2 + b^2}$)
عدد العمليات	**نوع العمليات**	**أسم العمليات**
١	ADD_R	ADD_R
٢	MUL_R	MUL_R
٣	MUL_R	MUL_R
٤	SQRT	SQRT

توضيح:

Network	العملية	المتغير	العملية	المتغير	أسم العملية
١	a x a	VD0 x VD0	a^2	VD20	عملية ضرب
٢	b x b	VD2 x VD2	b^2	VD22	عملية ضرب
٣	$a^2 + b^2$	VD20 + VD22	$a^2 + b^2$	VD40	عملية جمع
٤	$\sqrt{a^2 + b^2}$	SQRT(VD40)	$\sqrt{a^2 + b^2}$	VD4	جزر تربيعى

الشرح:

Network1:

بالضغط على I0.1 سوف يتم ضرب قيمة المتغير VD0 فى قيمة المتغير VD0 ويتم وضع الناتج فى المتغير VD20.

Network2:

بالضغط على I0.1 سوف يتم ضرب قيمة المتغير VD2 فى قيمة المتغير VD2 ويتم وضع الناتج فى المتغير VD22.

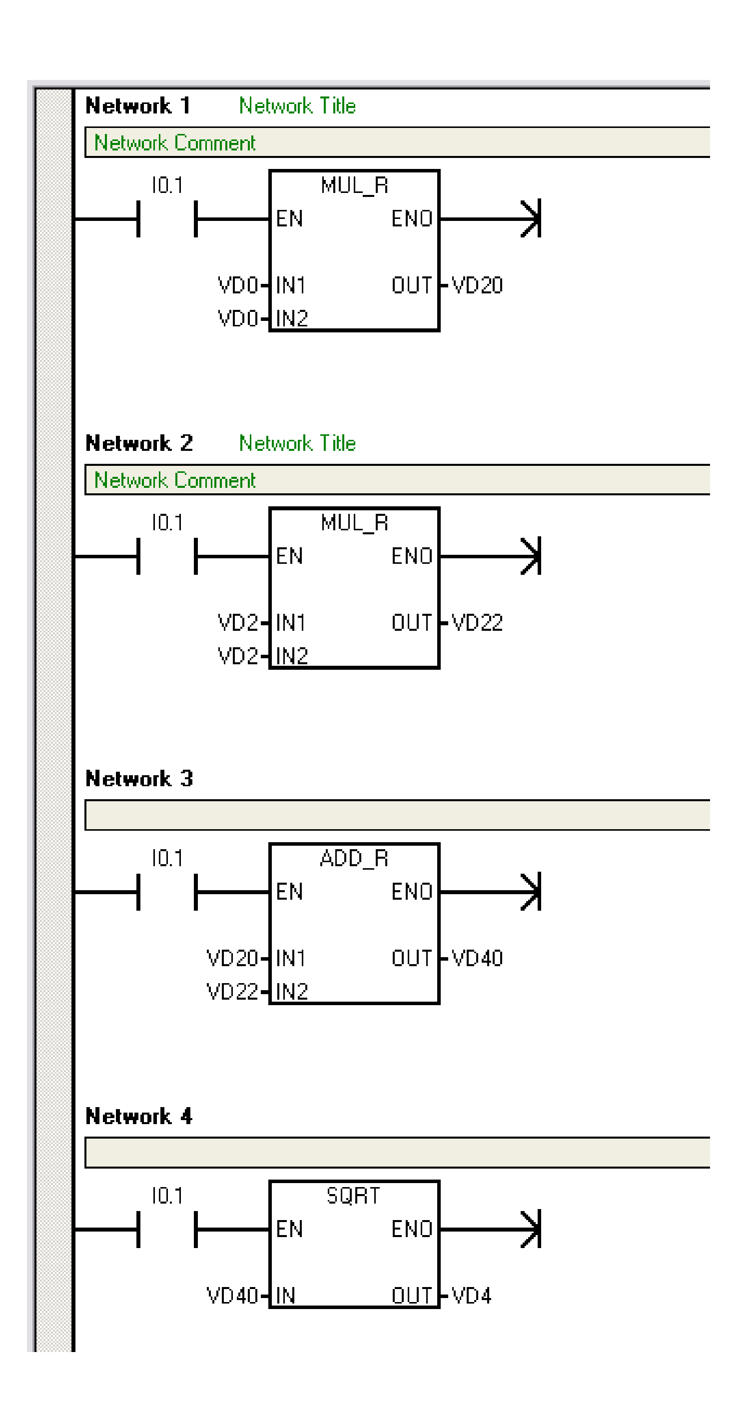
Network3:

بالضغط على I0.1 سوف يتم جمع قيمة المتغير VD20 وقيمة المتغير VD22 ويتم وضع الناتج فى المتغير VD40.

Network4:

بالضغط على I0.1 سوف يتم تطبيق الجذر التربيعى على قيمة المتغير VD40 ويتم وضع الناتج فى المتغير VD4.

البرنامج.



Network 1
Network Title
Network Comment
I0.1
MUL_R
EN
ENO
VD0
IN1
OUT
VD20
VD0
IN2
Network 2
Network Title
Network Comment
I0.1
MUL_R
EN
ENO
VD2
IN1
OUT
VD22
VD2
IN2
Network 3
I0.1
ADD_R
EN
ENO
VD20
IN1
OUT
VD40
VD22
IN2
Network 4
I0.1
SQRT
EN
ENO
VD40
IN
OUT
VD4

٢- قم بتنفيذ دائرة تحكم منطقية لمعادلة تقوم بتحويل الزاوية من Radiant إلى Degree.

علماً بأن المعادلة الخاصة بالتحويل هى.

$$D = \frac{G}{1.8} - 32$$

عدد الدخل	نوع الدخل	أسم الدخل
١	n.c.	I0.0/Stop
٢	n.o.	I0.1/Start
عدد المتغيرات	**نوع المتغيرات**	**أسم المتغيرات**
١	D.word	VD0(D)
٢	D.word	VD4(G)
٣	D.word	VD8(G/1.8)
عدد العمليات	**نوع العمليات**	**أسم العمليات**
١	ADD_R	ADD_R
٢	MUL_R	MUL_R
٣	MUL_R	MUL_R
٤	SQRT	SQRT

توضيح:

Network	العملية	المتغير	العملية	المتغير	أسم العملية
١	G/1.8	VD4 / 1.8	G/1.8	VD8	عملية قسمة
٢	(G/1.8) – 32	VD8 – 32	(G/1.8) – 32	VD0	عملية طرح

الشرح:

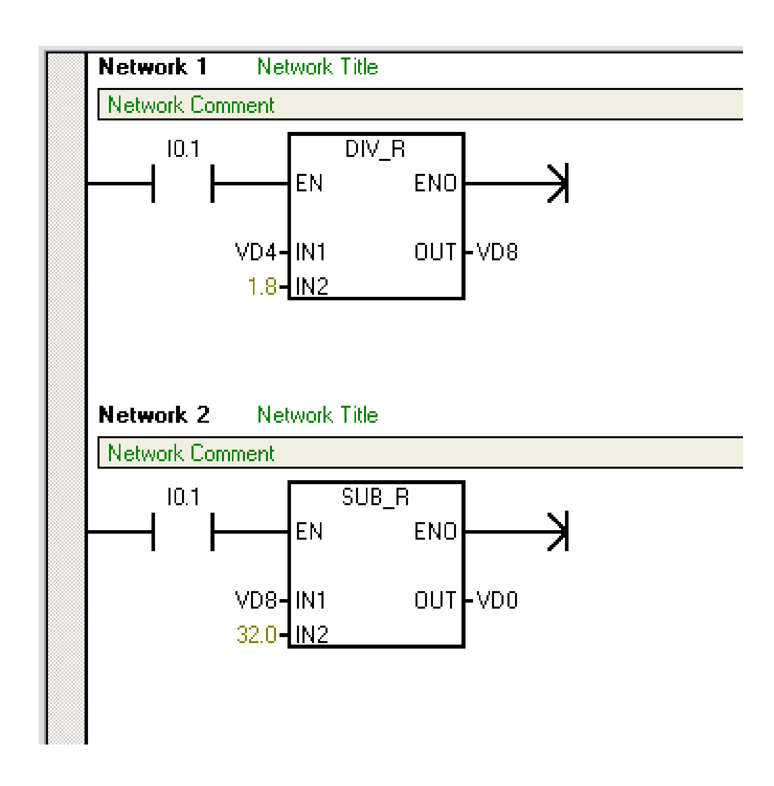
:Network1

بالضغط على I0.1 سوف يتم قسمة قيمة المتغير VD4 على ٨,١ ويتم وضع الناتج فى المتغير VD8.

:Network2

بالضغط على I0.1 سوف يتم طرح رقم ٣٢,٠ من قيمة المتغير VD8 ويتم وضع الناتج فى المتغير VD0.

البرنامج:



٣- قم بتنفيذ دائرة تحكم منطقية لمعادلة التالية:

$$Y = 2.5\,X^2 + \frac{3}{5}\,X - 4$$

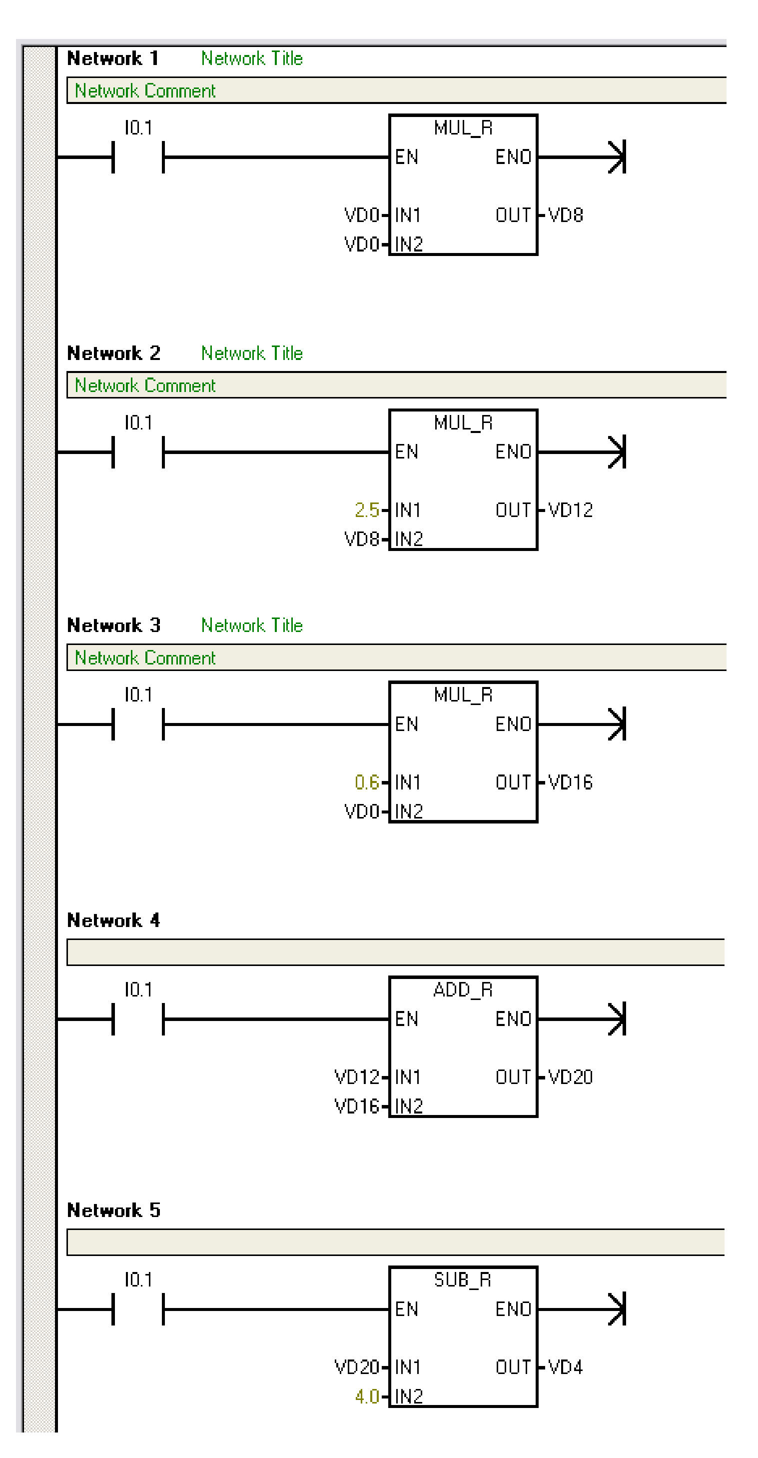
أسم الدخل	نوع الدخل	عدد الدخل
I0.1/Start	n.o.	١
أسم المتغيرات	**نوع المتغيرات**	**عدد المتغيرات**
VD0(x)	word	١
VD4(Y)	word	٢
أسم العمليات	**نوع العمليات**	**عدد العمليات**
ADD_R	ADD_R	١
MUL_R	MUL_R	٢
MUL_R	MUL_R	٣
SQRT	SQRT	٤

توضيح:

المتغير	العملية	المتغير	العملية	Network
VD8	x^2	VD0 x VD0	x . x	١
VD12	$2.5x^2$	2.5 x VD8	$2,5 . x^2$	٢
VD16	0.6x	0.6 x VD0	0,6 . x	٣
VD20	$2.5x^2+0.6x$	VD12 + VD16	$2,5 . x^2 + 0,6 . x$	٤
VD4	$2.5x^2+0.6x-4$	VD20 – 4.0	$2,5 . x^2 + 0,6 . x - 4$	٥

عملية جمع	**Network4**	عملية ضرب	**Network1**
عملية طرح	**Network5**	عملية ضرب	**Network2**
		عملية ضرب	**Network3**

البرنامج:


Network 1 Network Title
Network Comment
I0.1
MUL_R
EN ENO
VD0 IN1 OUT VD8
VD0 IN2
Network 2 Network Title
Network Comment
I0.1
MUL_R
EN ENO
2.5 IN1 OUT VD12
VD8 IN2
Network 3 Network Title
Network Comment
I0.1
MUL_R
EN ENO
0.6 IN1 OUT VD16
VD0 IN2
Network 4
I0.1
ADD_R
EN ENO
VD12 IN1 OUT VD20
VD16 IN2
Network 5
I0.1
SUB_R
EN ENO
VD20 IN1 OUT VD4
4.0 IN2

الشرح:

:Network1

بالضغط على I0.1 سوف يتم ضرب قيمة المتغير VD0 فى قيمة المتغير VD0 ويتم وضع الناتج فى المتغير VD8.

:Network2

بالضغط على I0.1 سوف يتم ضرب رقم ٥,٢ فى قيمة المتغير VD8 ويتم وضع الناتج فى المتغير VD12.

:Network3

بالضغط على I0.1 سوف يتم ضرب رقم ٦,٠ فى قيمة المتغير VD0 ويتم وضع الناتج فى المتغير VD16.

:Network4

بالضغط على I0.1 سوف يتم جمع قيمة المتغير VD12 فى قيمة المتغير VD16 ويتم وضع الناتج فى المتغير VD20.

:Network5

بالضغط على I0.1 سوف يتم طرح رقم ٤ من قيمة المتغير VD20 ويتم وضع الناتج فى المتغير VD4.

ملاحظة:

لقد تم تنفيذ المعادلة السابقة كمثال عابر ولكن فى حقيقة الأمر قد تكون هذه المعادلة خاصة بقيمة تناظرية تشير إلى درجة حرارة, سرعة محرك, شدة الضغط, إلخ....

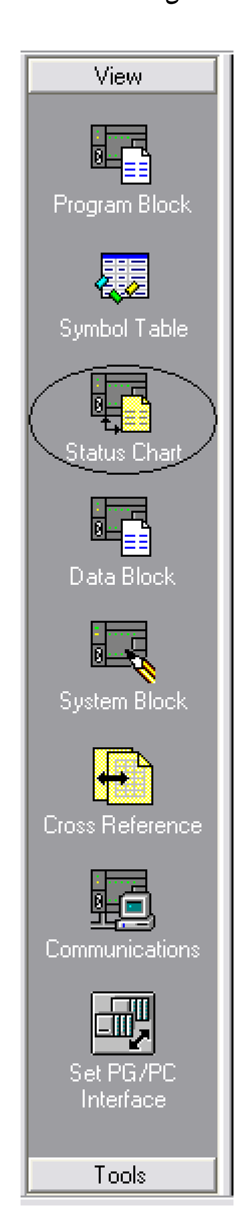
الباب الثانى

جدول الحالات

- محتويات جدول الحـــــــــــالات.
- أستخدامات جدول الحـــــــــــالات.
- المفاتيح المستخدمة فى جدول الحالات.
- الطرق المستخدمة فى أظهار حالة العناوين.
- كيفية كتابة مجموعة عناويـــــــــن.
- أمـــــــــــــــــر Write all.
- أمـــــــــــــــــر Force.
- كيفية تطبيق أمر Write all على تمارين.
- كيفية تطبيق أمر Force على تمـــــارين.
- الرسم التخطيطى لأى عنوان فى البرنامـــج.

جدول الحالات Status Chart:

تستخدم صفحة "جدول الحالات" الـ Status Chart لمعرفة الحالة الخاصة بكل عنوان من العناوين المستخدمة أو غير المستخدمة في البرنامج و يمكن أيضاً تعديل حالة المدخلات, المخرجات, الريليهات, المؤقتات الزمنية, العدادات و المتغيرات بواسطة هذه الصفحة.



طريقه استخدام صفحة "جدول الحالات ":

- بواسطة صفحة "جدول الحالات" الـ Status Chart يمكن معرفة حالة المدخلات و المخرجات وأى عنوان أخر دون الذهاب إلى مكان المداخل أو مكان المخرجات, فمثلاً قد يوجد من ضن مجموعة المفاتيح (حساس موصل أسفل مكينة معينة) فبدلاً من الذهاب للبحث عن المفتاح لمعرفة إذا كان مفتوح أو مغلق فيمكن معرفة الحالة بواسطة صفحة "جدول الحالات" بكل سهولة.

تستخدم أيضاً صفحة "جدول الحالات" الـ Status Chart لمعرفة الأعطال:

- فمثلا فى حالة أن كان المفتاح مغلق بينما تقول صفحة "جدول الحالات" أنة مفتوح فهذا يعنى أن المفتاح به مشكلة (ليس موصل بالكهرباء – تم التوصيل على العزل – الكهرباء الموصلة بالمفتاح أقل من الحد المسموح بة – مشكلة بنقاط التلامس الخاصة بالمفتاح – إلخ..).
- بينما فى حاله إذا كان الخرج (المحرك) لا يعمل بينما تشير صفحة جدول الحالات أن الخرج يعمل فهذا يشير إلى عطل غير مسئول عنه وحدة البرمجة نفسها (قد لا توجد تغذية لدائرة القوة – مشكلة بالريليه الميكانيكى أو بنقطة الريلية الذى يعمل كحماية بين وحدة البرمجة ودائرة القوى – نقاط القاطع الحرارى مفتوحة).

ملاحظة. سوف يتم شرح الأعطال بالتفصيل بجميع أنوعها فى الكتاب التالى.

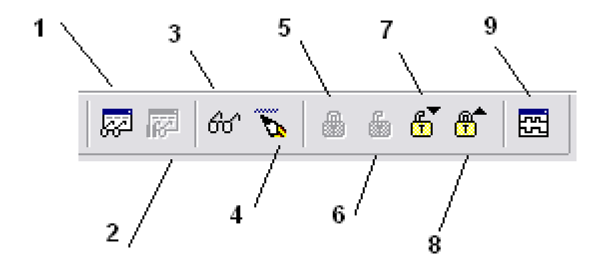
الشكل العام لصفحة **"جدول الحالات"**:

	Address	Format	Current Value	New Value
1	I0.0	Bit		
2	Q0.3	Bit		
3	T32	Bit		
4	C11	Bit		
5	T32	Signed		
6	C11	Signed		

المقصود بكلمة (Address – Format – Current value – New value)

- Address: حيث تكتب العناوين وليس بالشرط أن يكون العنوان مستخدم فى البرنامج.
- Format: حيث يتم أختيار الطريقة المراد أظهار بها حالة العنوان.
- Current value: حيث تظهر حالة العنوان.
- New value: حيث يمكن تعديل حالة العنوان.

مفاتيح هامة فى صفحة "جدول الحالات":



١- Chart status: لمعرفة حالة العناوين المكتوبة بصفة مستمرة.

٢- Pause chart status: لإيقاف عند نقطة معينة لمعرفة حالة العناوين المكتوبة.

٣- single read: لمعرفة حالة العناوين المكتوبة فى لحظة معينة.

٤- Write all: لتعديل أو تغير بعض العناوين ولكن حسب البرنامج اى حسب الشروط المستخدمة فى البرنامج, فمثلاً لا يمكن تشغيل محرك عكس حركة فى الاتجاهين معاً وهذا ليس ذكاء من جهاز ال PLC لأنه لا يعرف مقدماً أنه سوف يحدث قفله بل لأن شروط البرنامج تشير إلى أنه لا يمكن للاتجاهين أن يعمل معاً فى وقت واحد.

٥- **Force:** لتعديل أو تغير بعض العناوين مهما كان البرنامج اى أنه لا تؤخذ فى الاعتبار الشروط المستخدمة, فمثلاً يمكن تشغيل محرك عكس حركة فى الاتجاهين معاً لذالك يجب وضع الحمايات ليس فقط فى البرنامج بل على الريلية الميكانيكى إيضاً كم وضح فى الجزء الأول من هذا الكتاب.

٦- Unforce: لإلغاء أمر Force بالنسبة لـ bit اى لدخل واحد أو لخرج واحد فقط.

٧- Unforce all: لإلغاء أمر Force بالنسبة لـ byte أو word أو Dword أى لمجموعة مدخلات أو لمجموعة مخرجات.

٨– Read all forced: لمعرفة كل العناوين التى طبق عليها أمر Force خاصاً عندما تتعامل مع الجهاز لأول مرة يجب التأكد من أن كل المخارج تعمل بطريقة طبيعية وليس تحت تأثير أمر force.

٩– Trend view: بالضغط عليها يقوم جهاز الـ PLC بعمل رسم تخطيطى diagram للعناوين المكتوبة بصفحة "جدول الحالات" سواء كانت العناوين الموجودة هى (مدخلات – مخرجات – مؤقتات زمنية – عدادات – متغيرات – ريليهات داخلية – إلخ.....).

طريقة أظهار حاله العنوان الـ format :

Format: هى الطريقة المراد أظهار بها حالة العنوان و تنقسم إلى:

- Unsigned - Signed - Hexadecimal - Binary - Bit -
- ASCII - Floating point

الشكل العام لكل format:

- **Bit**: 2#0 حيث رقم اثنان يشير أن اللغة المستخدمة هى Binary.
- **Binary**: 2#0000_0000 حيث رقم "اثنان" يشير أن اللغة المستخدمة هى Binary.
- **Hexadecimal**: 16#2A حيث رقم "ستة عشر" يشير أن اللغة المستخدمة هى Hexa.
- **Signed**: ٧٨٥٨+ أو ٥٩٣– حيث يحتوى الرقم على أشارة سواء موجبة أو سالبة.
- **Unsigned**: ٦٧١١ حيث لا يحتوى الرقم على أشارة موجبة أو سالبة.
- **Floating point**: ٢ E+ ٥٣٩, ٢+ حيث أن الرقم المكتوب هو رقم عشرى.
- **ASCII**: "S" أو "<" حيث يشير إلى الحالة باستخدام حروف أو رموز.

الشرح:

- بعض العناوين لا يمكن أن تظهر بأكثر من format, فمثلاً المفتاح I0.1 أو الخرج Q2.6 أو الريليه M1.1 أو المتغيرات V23.4 لا يمكن أن يكون لهم format غير الـ Bit وهذا لأن حالة الدخل أو الخرج أو الريليه أو المتغير أما أن تكون واحد أو صفر.

- يستخدم الـ format (Floating point) مع الأرقام العشرية, ومع الأرقام العشرية ومع المعدلات أو المتغيرات أو المداخل والمخارج التناظرية.

- يعتبر الـ format (ASCII) من الـ format الأقل استخداماً حيث يظهر الحالة الخاصة بالعنوان المذكور باستخدام أحرف أو رموز بدلاً من استخدام الأرقام.

- بعض العناوين يمكن أن تعمل مع أكثر من format, فمثلاً المؤقت الزمنى T32 أو العداد C0 أو المتغير VB4 يمكن أن يكون لهم format مختلفة مثل الـ Binary أو Signed أو Unsigned أو Hexadecimal أو ASCII ولكن يفضل دائماً اختيار الأفضل حسب نوع العنوان المستخدم.

- بالنسبة للمؤقت الزمنى أو للعداد يمكن تكرار العنوان مرتين بحيث أن تكون المرة الأولى مثلاً format (Bit) وهى تعنى الحالة الخاصة بنقطة العداد أو نقطة المؤقت الزمنى بحيث تشير أذا كانت مغلقة أو مفتوحة بينما المرة الثانية تكون مثلاً FORMAT (Signed, Unsigned, Binary أو Hexadecimal) بحيث تشير إلى قيمة الرقمية للعداد أو القيمة الرقمية للمؤقت الزمنى.

- فى حالة الاستعلام عن حالة مجموعة من المدخلات أو المخرجات فبدلاً من كتابة كل دخل أو كل خرج أو كل ريليه أو كل متغير على حدى يمكن الاستعلام عن مجموعة مكونة من ٨ أو ١٦ أو ٣٢ دخل أو خرج أو ريليه أو متغير معاً عن طريق حرف الـ B الذى يرمز إلى الـ Byte أى ثمانية

أو عن طريق حرف الـ W الذى يرمز إلى الـ Word أى ستة عشر أو عن طريق حرف الـ D الذى يرمز إلى الـ Dword أى اثنان و ثلاثون فمثلاً:

شرح مبسط

- IB0: تعنى أول "ثمانية مدخلات".
- IB1: تعنى ثانى "ثمانية مدخلات".
- QB6: تعنى سابع "ثمانية مخرجات".
- QB9: تعنى عاشر "ثمانية مخرجات ".
- IW2: تعنى ثانى "ستة عشر دخل".
- IW6: تعنى رابع "ستة عشر دخل ".
- QW4: تعنى ثلاث "ستة عشر خرج ".
- QW8: تعنى خامس "ستة عشر خرج ".
- ID0: تعنى أول " اثنان وثلاثين دخل ".
- ID4: تعنى ثانى " أثنان وثلاثين دخل ".
- QD8: تعنى ثلاث "اثنان وثلاثين مخرج".
- QD12: تعنى رابع "اثنان وثلاثين مخرج".

للتوضيح أنظر صفحة ٢٠١ فى الجزء الأول من هذا الكتاب.

باستخدام صفحة "جدول الحالات" الـ Status Chart التى تستخدم لمعرفة الحالة الخاصة بكل عنوان من العناوين المستخدمة فى البرنامج أو غير المستخدمة حتى, يمكن أيضاً تعديل أو تغيير حاله العناوين المستخدمة فى البرنامج وغير المستخدمة أيضاً فلذالك توجد طريقتان للتعديل أو التغير فى البرنامج:

الطريقة الأولى هى write all. 

الطريقة الثانية هى force: 

الطريقة الأولى للتعديل فى البرنامج.

تستخدم Write all فى تعديل البرنامج ولكن "**مع مراعاة الشروط**" المستخدمة فى البرنامج.
المقصود بكلمة **مراعاة الشروط** المستخدمة فى البرنامج أى أنه لا يمكن تنفيذ أى أمر بواسطة الـ write all وفى نفس الوقت يكون من غير الممكن تنفيذ هذا الأمر بواسطة البرنامج, فمثلاً:

أ. لا يمكن لمحرك أن يعمل بينما يكون مفتاح الإيقاف الخاص بالمحرك مفتوح.

ب. لا يمكن لمحرك أن يقف بينما يكون مفتاح التشغيل الخاص بالمحرك مغلق.

ت. لا يمكن لمحرك عكس حركة (يعمل فى اتجاهين) أن يعمل فى الاتجاهين معاً فى نفس الوقت.

ث. لا يمكن لمؤقت زمنى أن يبدأ بالعمل بينما لا تكون هناك تغذية.

ج. لا يمكن لعداد أن يبدأ بالعد بينما تكون كل النقاط الخاصة بالإشارة غير مغلقة.

فلذالك ولا حتى أمر write all يستطيع أن يقوم بتنفيذ أى من النقاط السابقة

الطريقة:

١ – تتم كتابة العناوين المراد التعامل معها سواء كانت, مخرجات أو مؤقتات زمنية أو عدادات أو متغيرات إلخ.....

	Address
1	I0.1
2	Q0.4
3	T32
4	C14
5	VW0

٢– يتم اختيار الطريقة format المراد إظهار بها حالة العنوان المذكور أعلاه.

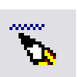
Format
Bit
Binary
Signed
Unsigned
Hexadecimal
ASCII

٣– يتم الضغط على مفتاح chart status لمعرفة حالة العناوين بصفة مستمرة.

Current Value
2#1
2#1
+11103
+2
+30000

٤ – يكتب فى new value مقابل كل عنوان التعديل المراد تنفيذه.

New Value
2#0
2#1
+1000
+30
+0

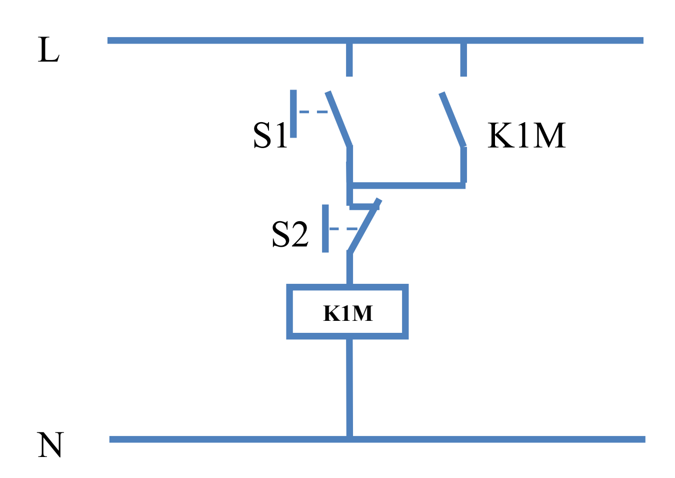
بعد ذالك يتم الضغط على write all  **لتنفيذ التغيير المراد تطبيقه**

ملاحظة:

- لا يمكن تعديل حالة المدخلات باستخدام أمر Write all.
- فى حالة تعديل خرج يكتب واحد للتشغيل أو صفر للإيقاف.
- فى حالة تعديل قيمة العداد تكتب القيمة المراد العداد الوصول إليها وليس بالضرورى أن يتم تنفيذ هذا التعديل أثناء عمل العداد.
- فى حالة تعديل قيمة المؤقت الزمنى تكتب القيمة المراد الوصول إليها بواسطة المؤقت الزمنى ولكن بشرط أن يكون تنفيذ هذا التعديل أثناء عمل المؤقت الزمنى.
- فى حالة كتابة أى قيمة على متغيرات يجب أن يأخذ فى الاعتبار بأن القيمة المراد كتابتها بواسطة أمر write all يجب أن لا تكون أكبر من أكبر رقم يمكن كتابته على المتغيرات وينطبق هذا على جميع المتغيرات بمختلف أحجامها حيث أن فى حاله كتابة قيم كبيرة على متغيرات صغيرة قد يسبب هذا مشاكل سوف يتم توضيحها فى الكتاب التالى الخاص بالأعطال والتمرين العملية.

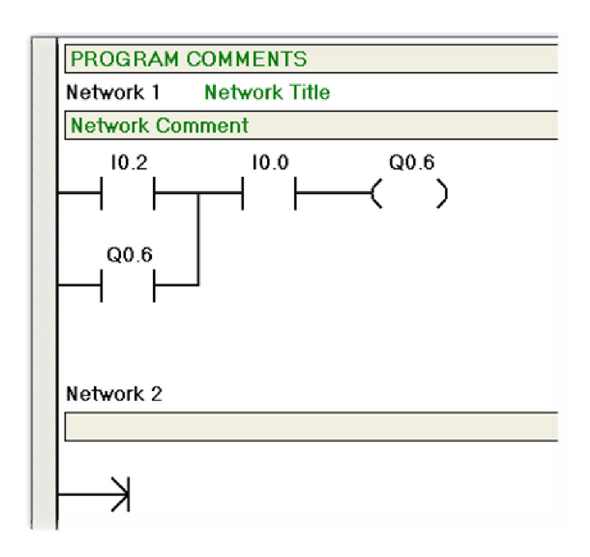
مثال باستخدام أمر Write all:

محرك يعمل من مكان واحد.



عدد الدخل	نوع الدخل	أسم الدخل
١	n.o.	I0.2/S1
٢	n.c.	I0.0/S2
عدد الخرج	**نوع الخرج**	**أسم الخرج**
١	كونتكتور	Q0.6/K1M

البرنامج:



الخطوات:

- أولاً: يكتب العنوان المراد التعامل معه فى Address و فى هذا التمرين العنوان هو Q0.6
- ثانياً: يتم اختيار ال Format حسب ال Address و فى هذه التمرين ال Format هو Bit فقط.
- ثالثاً: تكتب الحالة المراد تعديلها أى 2#0 للإيقاف و 2#1 للتشغيل.
- رابعاً: يتم الضغط على Write all لتنفيذ التعديل المرغوب فيه.

	Address	Format	Current Value	New Value
1	Q0.6	Bit		2#1

ملاحظة:

لإلغاء أمر Write all الذى تم استخدامه فى المثال السابق توجد طريقتان:

١- التحكم بالخرج عن طريق البرنامج أى بفتح I0.0 وهذا لأن أمر write all لا يستطيع أن يخالف البرنامج نفسه من حيث طريقة التحكم بالخرج.

٢- التحكم بالخرج عن طريق Status chart أى للإيقاف يكتب 2#0 و للتشغيل يكتب 2#1 ثم الضغط على Write all مرة أخرى.

	Address	Format	Current Value	New Value
1	Q0.6	Bit		2#0

يمكن أيضاً تغير قيم المؤقتات الزمنية بواسطة نفس الأمر:

الخطوات:

- **أولاً:** يكتب العنوان المراد التعامل معه فى Address و فى هذه الحالة العنوان هو T32.
- **ثانياً:** يتم اختيار الـ Format حسب الـ Address و فى هذه الحالة الـ Format هو Bit بالنسبة لنقطة المؤقت الزمنى و Signed بالنسبة للقيمة الفعلية للمؤقت الزمنى.
- **ثالثاً:** تكتب الحالة المراد تعديلها أى لتغيير القيمة إلى أربع ثوانى تكتب قيمة ٤٠٠٠.
- **رابعاً:** يتم الضغط على Write all لتنفيذ التعديل المرغوب فيه.

بشرط أن يتم هذا التطبيق أثناء عمل المؤقت الزمنى لأننا نتحدث عن زمن حقيقى فلذالك ليس من المنطقى أن يتم أعطاء أى زمن للمؤقت ثم يتم بدء حسب هذا مؤخراً عند تشغيل المؤقت الزمنى فهذا لأننا نتحدث عن زمن حقيقى real time.

	Address	Format	Current Value	New Value
1	T32	Signed		+4000

يمكن أيضاً تغير قيم العدادات:

الخطوات:

- **أولاً**: يكتب العنوان المراد التعامل معه فى Address و فى هذه الحالة العنوان هو C10.
- **ثانياً**: يتم اختيار الـ Format حسب الـ Address و فى هذه الحالة الـ Format هو Bit بالنسبة لنقطة المؤقت الزمنى و Signed بالنسبة للقيمة الفعلية للعداد.
- **ثالثاً**: تكتب الحالة المراد تعديلها أى لتغيير القيمة إلى خمسة عشر تكتب قيمة ١٥.
- **رابعاً**: يتم الضغط على Write all لتنفيذ التعديل المرغوب فيه.

	Address	Format	Current Value	New Value
1	C10	Signed		+15

ملاحظة:

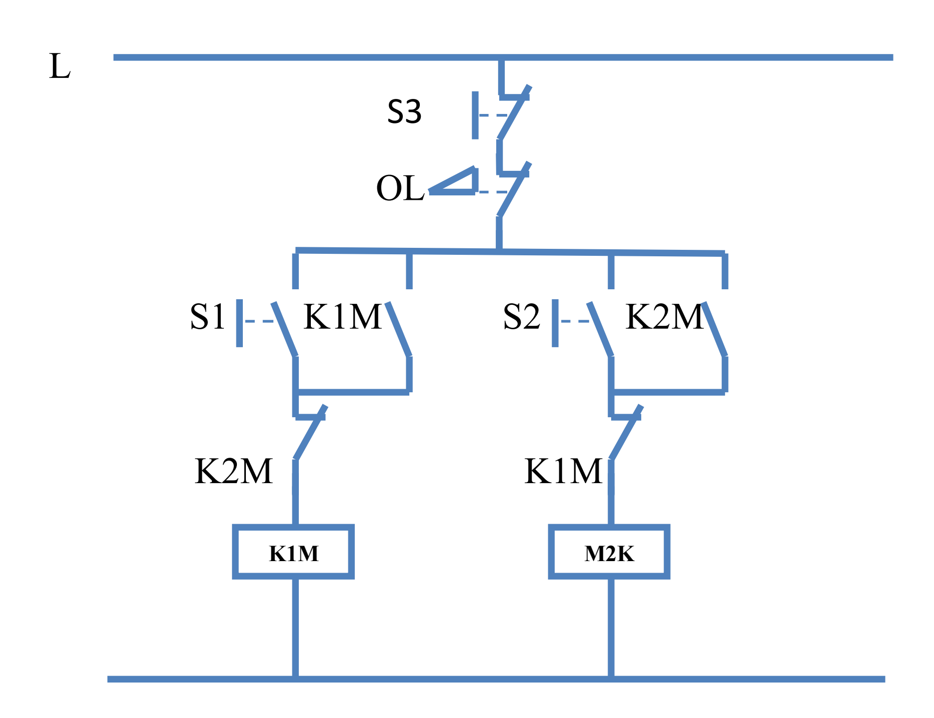
- يجب دائماً الضغط على Write all لتنفيذ القيم المكتوبة فى New value.

- ليس بالشرط أن يتم هذا التطبيق أثناء عمل العداد لأننا نتحدث عن إشارات حقيقية تحتسب فقط عند إرسال الإشارة بواسطة الضغط على المفتاح مثلاً فلذالك من المنطقى جداً أن يتم أعطاء أى رقم للعداد ثم يتم بدء حسب هذا مؤخراً عند تشغيل العداد أو فى الحال إذ كان العداد يعمل أساساً فهذه العملية لا ترتبط بزمن حقيقى real time.

- فى حالة إرسال قيمة إلى العداد حتى وأن كانت هذه القيمة أكبر أو تساوى القيمة المسبقة (إى القيمة المقصودة) فأنه يشترط دائما أن يعمل العداد حتى يتم تفعيل الحاله الحقيقية للنقاط الخاصة بالعداد لأنه ليس من المنطقي أن يغير العداد النقاط الخاصة به أن كان لم يعمل بعد.

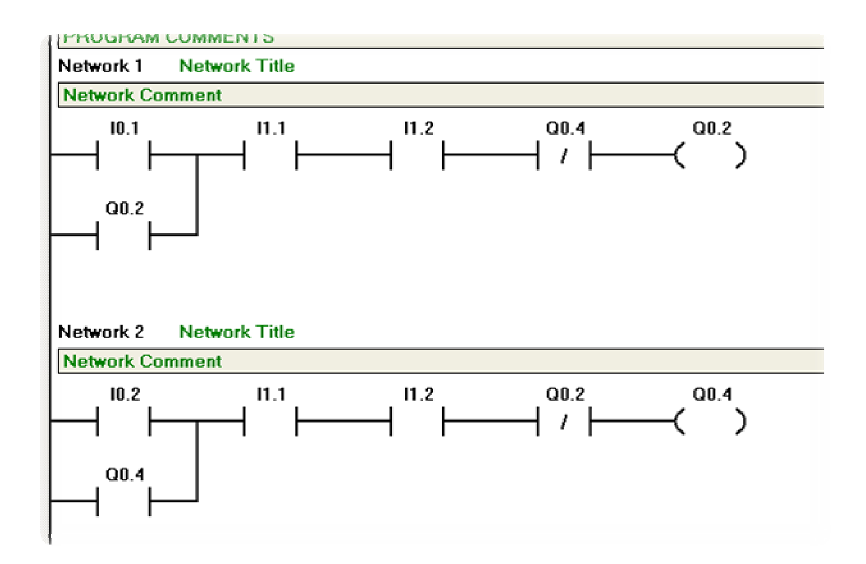
مثال أخر باستخدام أمر **Write all**:

محرك يعمل فى اتجاهين:



عدد الدخل	نوع الدخل	أسم الدخل
١	n.o.	I0.1 / S1
٢	n.o.	I0.2 / S2
٣	n.c.	I1.1 / S3
٤	n.c.	I1.2 / OL
عدد الخرج	**نوع الخرج**	**أسم الخرج**
١	كونتكتور	Q0.2 / K1M
٢	كونتكتور	Q0.4 / K2M

البرنامج



الخطوات:

- **أولاً:** يكتب العنوان المراد التعامل معه فى Address و فى هذا التمرين العنوان هو Q0.2 و Q0.4
- **ثانياً:** يتم اختيار الـ Format حسب الـ Address و فى هذه التمرين الـ Format هو Bit فقط.
- **ثالثاً:** تكتب الحالة المراد تعديلها أى 2#0 للإيقاف أو 2#1 للتشغيل بالنسبة لأى من الخرجين.
- **رابعاً:** يتم الضغط على Write all لتنفيذ التعديل المرغوب فيه.

	Address	Format	Current Value	New Value
1	Q0.2	Bit		2#0
2	Q0.4	Bit		2#1

ملاحظة:

- لإلغاء أمر Write all الذى تم استخدامه فى المثال السابق تتم استخدام نفس الطريقتان الذين سبق شرحهم.
- من الشىء المهم جداً فى الـ Write all أنه لا يمكن أن يقوم بتغير أى حالة دون مراعاة الشروط الموجودة فى البرنامج, فمثلاً التمرين السابق هو عبارة عن محرك اتجاهين ومن المعروف أنة لا يمكن تشغيل اتجاهين معاً لأن التمرين يحتوى على نقط مغلقة من الطرفين فلذلك ولا حتى أمر Write all يستطيع أن يقوم بتشغيل الاتجاهين معاً.

الطريقة الثانية للتعديل فى البرنامج.

تستخدم Force فى تعديل البرنامج **"وبدون أى مراعاة للشروط"** المستخدمة فى البرنامج.
المقصود بكلمة **بدون مراعاة الشروط** المستخدمة فى البرنامج أى أنه يمكن تنفيذ أى أمر بواسطة الـ force وفى نفس الوقت يكون هذا الأمر غير ممكن تنفيذه بواسطة البرنامج فمثلاً:

أ. يمكن لمحرك أن يعمل بينما يكون مفتاح الإيقاف الخاص بالمحرك مفتوح.
ب. يمكن لمحرك أن يقف بينما يكون مفتاح التشغيل الخاص بالمحرك مغلق.
ت. يمكن لمحرك عكس حركة أن يعمل فى الاتجاهين معاً فى نفس الوقت (قفله).

الطريقة:

١ – تتم كتابة العناوين المراد التعامل معها سواء كانت, مداخل, مخارج, الخ........

Address

٢– يتم اختيار الطريقة format المراد أظهار بها حالة العنوان المذكور أعلاه كما وضح فى ما قبل فى نفس الباب.

Format
Bit
Binary
Signed
Unsigned
Hexadecimal
ASCII

٣– يتم الضغط على chart status لمعرفة حالة العناوين المكتوبة بصفة مستمرة.

Current Value
2#1
2#1
+11103
+2
+30000

٤– يكتب مقابل العنوان new value التعديل المراد تنفيذه.

New Value
2#0
2#1
+1000
+30
+0

ملاحظة:

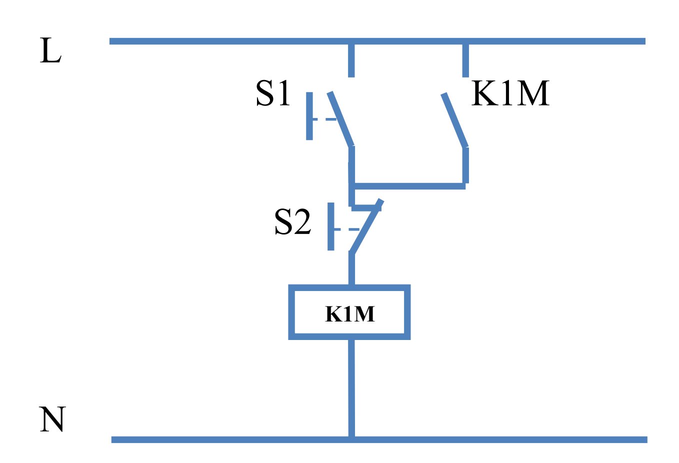
- يمكن تعديل حالة الدخل باستخدام أمر Force.
- فى حالة تعديل خرج يكتب واحد للتشغيل أو صفر للإيقاف.
- لا يمكن تعديل قيمة العداد أو المؤقت الزمنى باستخدام أمر Force.

- في حاله تغيير أى خرج إلى 2#1 لا يعني أن إلغاء الحالة سيكون عن طريق 2#0 لأن في الحالتين لا تراعى الأولويات أو الشروط الخاصة بالبرنامج.

بعد ذالك يتم الضغط على Force.
لإلغاء أمر Force بالنسبة لعنوان واحد فأنة يتم الضغط على Unforce.
لإلغاء أمر Force بالنسبة لكل العناوين فأنة يتم الضغط على Unforce all.
لمعرفة أى العناوين التي تم تطبيق أمر Force عليها فأنة يتم الضغط على Read all forced.

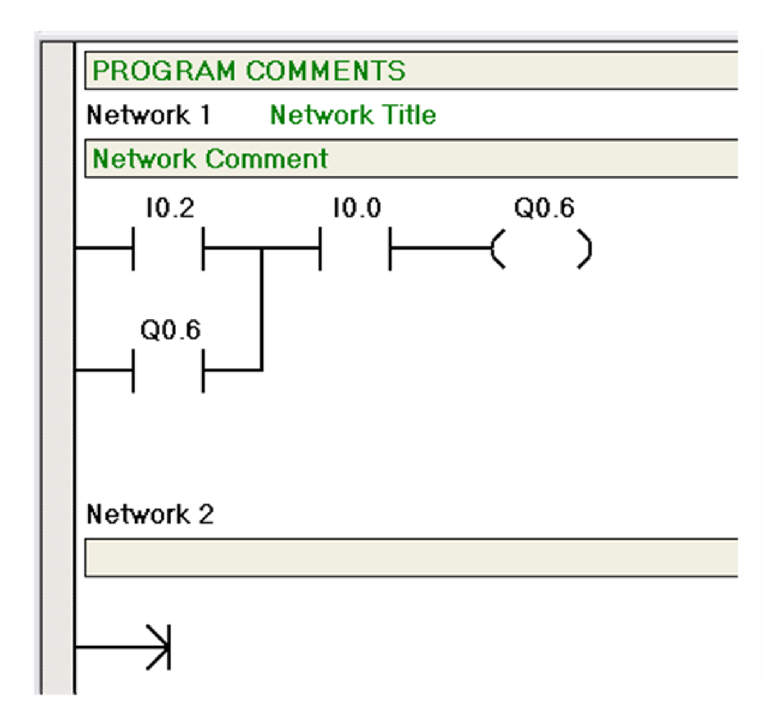
مثال باستخدام أمر Force:

محرك يعمل من مكان واحد.



عدد الدخل	نوع الدخل	أسم الدخل
١	n.o.	I0.2/S1
٢	n.c.	I0.0/S2
عدد الخرج	**نوع الخرج**	**أسم الخرج**
١	كونتكتور	Q0.6/K1M

البرنامج



الخطوات:

- **أولاً**: يكتب العنوان المراد التعامل معه فى address و فى هذا التمرين العنوان هو Q0.6
- **ثانياً**: يتم اختيار الـ format حسب الـ address و فى هذه التمرين الـ format هو bit فقط.
- **ثالثاً**: تكتب الحالة المراد تعديلها أى 2#0 للإيقاف أو 2#1 للتشغيل.
- **رابعاً**: يتم الضغط على force لتنفيذ التعديل المرغوب فيه.

	Address	Format	Current Value	New Value
1	Q0.6	Bit		2#1

ملاحظة:

لإلغاء أمر Force الذى تم استخدامه فى المثال السابق توجد طريق واحدة فقط وهى:

- لإلغاء أمر Force يستخدم أمر Unforce دون كتابة أى شيء.
- أمر Force بقيمة 2#0 ليس عكسها Force بقيمة 2#1 و العكس صحيح أيضاً ولكن لإلغاء أمر Force دائماً وأبداً يستخدم أمر Unforce.

	Address	Format	Current Value	New Value
1	Q0.6	Bit		2#0

- لا يمكن تغير قيم المؤقتات الزمنية باستخدام force:

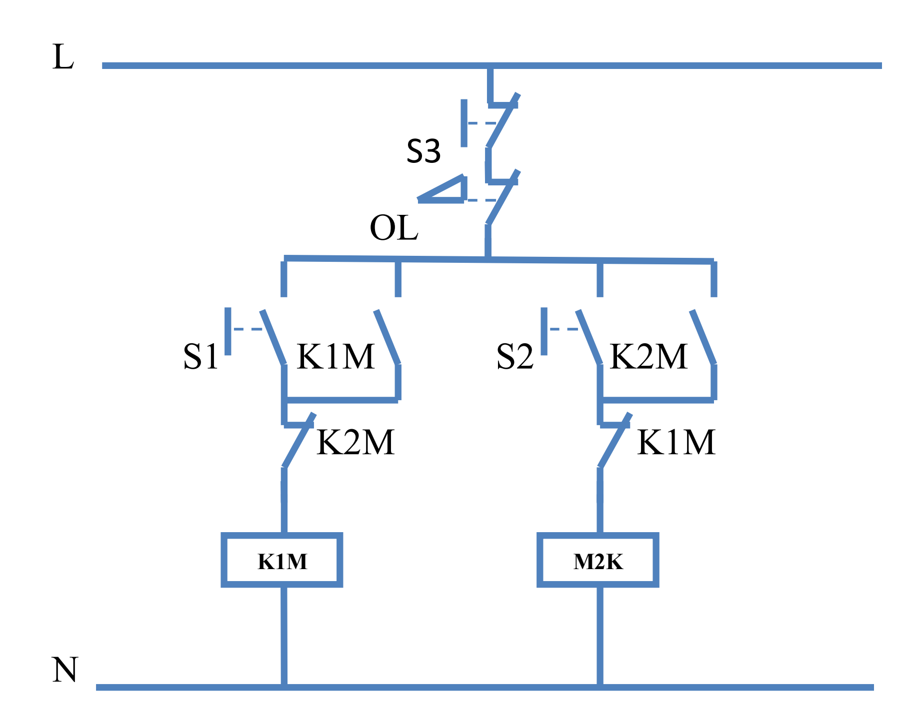
	Address	Format	Current Value	New Value
1	T32	Signed		+4000

- لا يمكن تغير قيم العدادات باستخدام force:

	Address	Format	Current Value	New Value
1	C10	Signed		+15

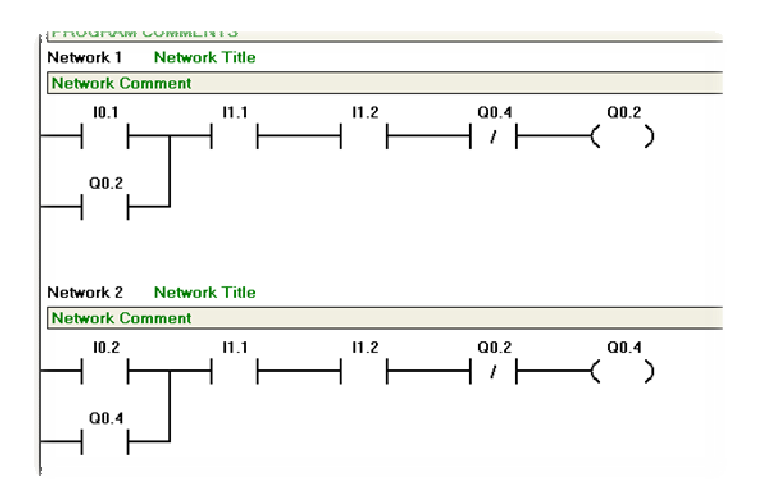
مثال أخر باستخدام أمر **Force**:

محرك يعمل فى اتجاهين:



عدد الدخل	نوع الدخل	أسم الدخل
١	n.o.	I0.1 / S1
٢	n.o.	I0.2 / S2
٣	n.c.	I1.1 / S3
٤	n.c.	I1.2 / OL
عدد الخرج	**نوع الخرج**	**أسم الخرج**
١	كونتكتور	Q0.2 / K1M
٢	كونتكتور	Q0.4 / K2M

البرنامج:



الخطوات:

- **أولاً:** يكتب العنوان المراد التعامل معه فى address و فى هذا التمرين العنوان هو Q0.2 و Q0.4
- **ثانياً:** يتم اختيار الـ format حسب الـ address و فى هذا التمرين الـ format هو bit فقط.
- **ثالثاً:** تكتب الحالة المراد تعديلها أى 2#0 للإيقاف أو 2#1 للتشغيل بالنسبة للخرجين.
- **رابعاً:** يتم الضغط على force لتنفيذ التعديل المرغوب فيه.

	Address	Format	Current Value	New Value
1	Q0.2	Bit	2#0	2#1
2	Q0.4	Bit	2#1	2#0

ملاحظة:

- لإلغاء أمر Force الذى تم استخدامه فى المثال السابق تتم استخدام نفس الطريقة التى تم شرحها فى ما سبق.

- من الشيء المهم جداً فى ال force أنه يمكن أن يقوم بتغير أى حالة دون مراعاة الشروط الموجودة فى البرنامج, فمثلاً التمرين السابق هو عبارة عن محرك اتجاهين ومن المعروف أنة لا يمكن تشغيل اتجاهين معاً لأن التمرين يحتوى على نقط مغلقة من الطرفين ولكن يستطيع أمر force أن يقوم بتشغيل الاتجاهين معاً وهذا يعنى أنه ستحدث "قفلة" على أطراف المحرك.

- أمر force يستطيع تغير قيمة الدخل أيضاً.

- نظراً إلى أن أمر force من الأوامر التى أذا أنسيت قد تسبب المشاكل فأنة تظهر علامة قفل مغلق بالقرب من العنوان الذى تم تطبيق أمر force عليه.

- أذا تم تطبيق أمر force بقيمة 2#1 على الخرج Q0.0 فأنه سيعمل فقط هذا الخرج, أى أنة أذا كان البرنامج قد صمم بحيث عندما يعمل هذا الخرج يعمل أيضاً معه مؤقت زمنى فأنة لن سيعمل لأن الخرج لم يعمل بطريقة طبيعية بل عمل عن بواسطة أمر force.

يمكن استخدام أمر force فى حل بعض الأعطال, فمثلاً:

- أذا كان المحرك لا يعمل فيمكن بدايتاً تطبيق أمر Force على كل مفتاح من المفاتيح التى تتحكم بالمحرك على حدا بحيث أنه قد يكون هناك مشكلة فى دخل معين فإذا عمل المحرك عند تنفيذ أمر force على المفتاح I2.3 مثلاً فهذا يعنى أنه توجد مشكلة بتلك المفتاح.

أمر ال.رسم التخطيطى Trend view:

يعتبر هذا الأمر من العماليات التوضيحية الموجودة بالبرنامج حيث يقوم بعمل "رسم تخطيطى" لكل العناوين المذكورة فى صفحة Status chart.

الخطوات:

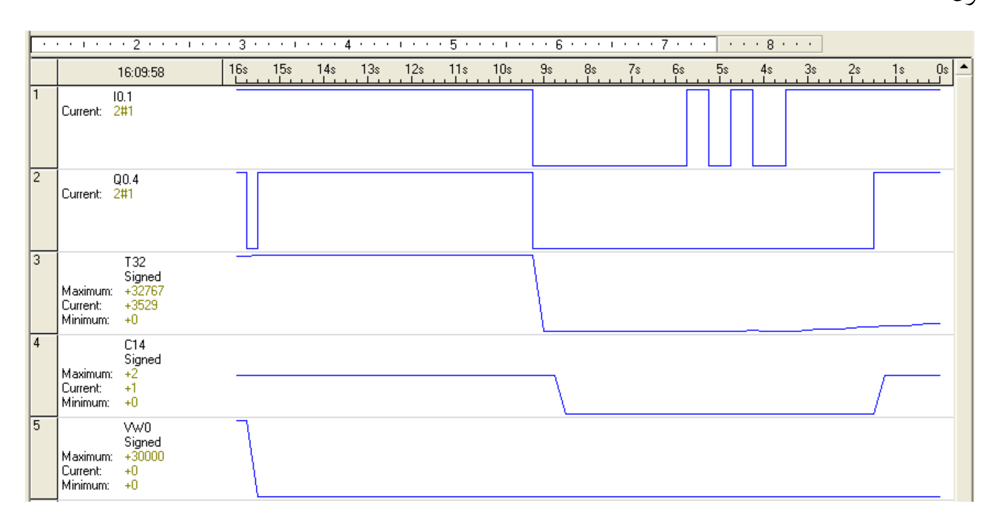
١ – تكتب العناوين المراد إظهارها فى "الرسم التخطيطى" فى جدول الحالات.

	Address	Format	Current Value	New Value
1	I0.1	Bit		
2	Q0.4	Bit		
3	T32	Signed		
4	C14	Signed		
5	VW0	Signed		

٢– يتم الضغط على chart status لمعرفة القيمة الحالية لكل عنوان من الخمسة عناوين المذكورة فى الجدول.

	Address	Format	Current Value	New Value
1	I0.1	Bit	2#1	
2	Q0.4	Bit	2#1	
3	T32	Signed	+32767	
4	C14	Signed	+2	
5	VW0	Signed	+30000	

٣– يتم الضغط على trend view لرأيه الرسم التخطيطى لكل عنوان من الخمسة عناوين المذكورة فى الجدول.

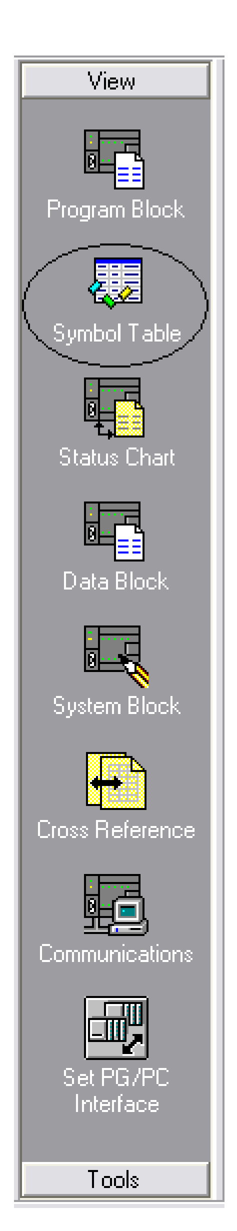


الباب الثالث

جدول الرموز

- **شرح جـــــــــــــــــــــدول الرموز.**
- **محتويات جـــــــــــــــــــدول الرموز.**
- **المفاتيح المستخدمة فى صـــــفحة الرموز.**
- **الأخـــــــــــــطاء الممكن التعرض لها.**
- **التعليقات الخاصة بصفحة جـــدول الرموز.**
- **جدول الرموز الخاص بـــصفحات البرمجة.**
- **طريقة البرمـــــــــــجة باستخدام العناوين.**
- **طريقة البرمجة باستخدام الرمـــــــــــوز.**
- **جـــــــــدول الـ S7-200 symbols.**

جدول الرموز symbol table :



تستخدم صفحة "جدول الرموز" الـ Symbol Table لكى يكون لكل عنوان من العناوين المستخدمة فى البرنامج رمز و يمكن أيضاً كتابة تعليقات خاصة بكل رمز حتى يصبح من السهل على المبرمج التعرف على وظيفة كل مفتاح, ريليه, خرج, عداد, مؤقت زمنى, إلخ.

طريقة استخدام صفحة "جدول الرموز":

- يمكن عمل البرنامج أولاً ثم إعطاء رموز لكل عنوان بواسطة صفحة "جدول الرموز".
- يمكن كتابة الرموز أولاً فى صفحة "جدول الرموز" ثم عمل البرنامج حسب الرموز المعطاة أو المحددة من قبل.

الأخطاء الممكن حدوثها:

١-كتابة أى رمز يكون بالصدفة أسم من أسماء العمليات المتعارف عليها فى البرمجة.

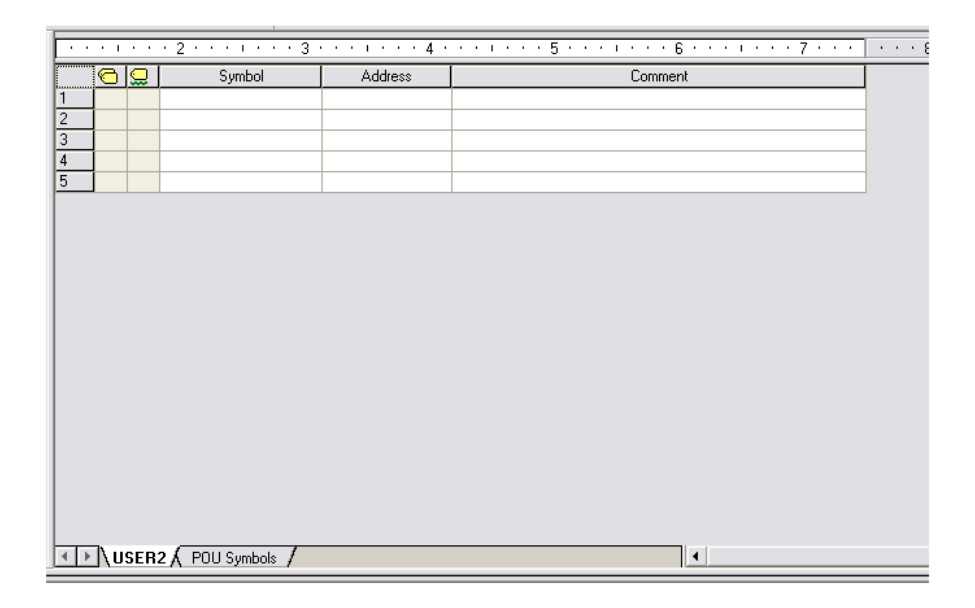
- أى أنه لا يمكن كتابة كلمة ("stop", "S", "R", "JMP", "LBL", "END", "ENI", "TON", "TOF", "TONR", "CTU", "CTD", "CTUD") لأن كل هذه الكلمات هى دوال لعمليات تستخدم فى البرمجة كما سيتم شرحها فيما بعد.

٢-كتابة أى رمز مكون من كلمتين و يكون بينهم مسافة.

- أى أنه لا يمكن كتابة كلمة ("first button", "second button ", "motor right", "motor left").

٣- تكرار نفس الرمز أكثر من مرة.

- أى أنه لا يمكن كتابة أى رمز أكثر من مرة حتى ولو كان مع نفس العنوان.



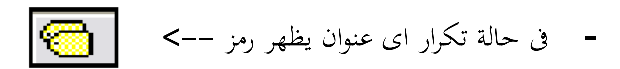
ملاحظة:

Symbol: حيث يكتب الرمز الخاص بكل عنوان بشرط عدم تكرار الرمز مع عناوين أخرى.

Address: حيث يكتب العنوان الخاص بالبرنامج بشرط أن يكون العنوان مستخدم فى البرنامج بالفعل.

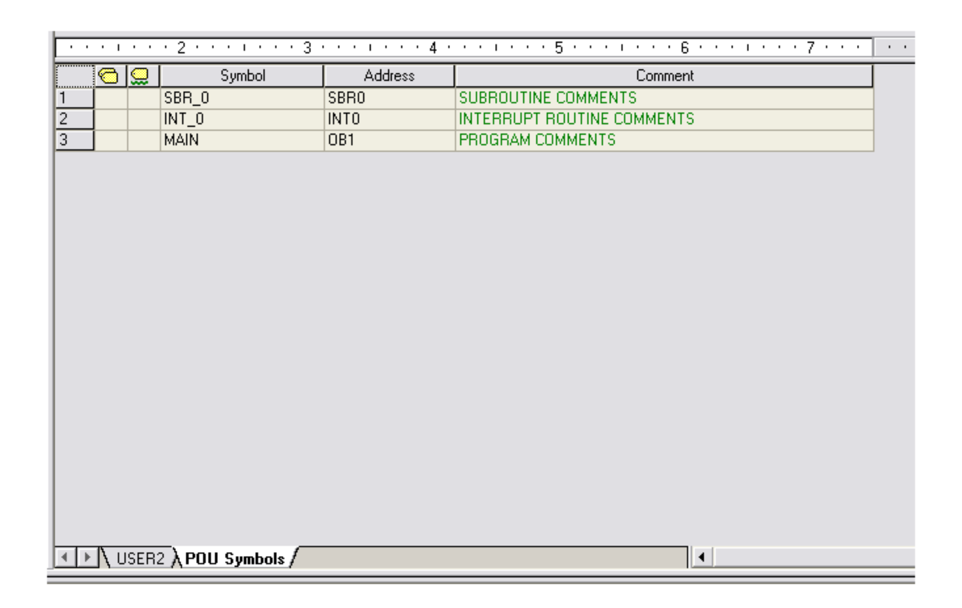
Comment: حيث تكتب الملاحظات الخاص بكل عنوان مستخدم فى البرنامج.

- فى حالة تكرار اى عنوان يظهر رمز -->



- فى حالة كتابة اى عنوان ليس مستخدم فى البرنامج يظهر رمز -->





			Symbol	Address	Comment
1			SBR_0	SBR0	SUBROUTINE COMMENTS
2			INT_0	INT0	INTERRUPT ROUTINE COMMENTS
3			MAIN	OB1	PROGRAM COMMENTS

ملاحظة:

Symbol: حيث توجد الرموز الخاصة بالثلاث أماكن الخاصة بصفحات البرمجة.

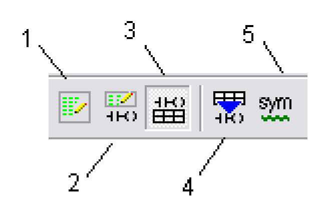
Address: حيث توجد العناوين الخاصة بالثلاث أماكن الخاصة بصفحات البرمجة.

Comment: حيث توجد الملاحظات الخاصة بالثلاث أماكن الخاصة بصفحات البرمجة.

ملاحظة:

- لا يمكن تعديل الرمز أو العنوان أو التعليقات الخاصة بصفحات البرمجة الثلاثة.
- فى حالة إضافة صفح أخرى خاصة بالبرمجة يضاف الرمز الخاص بها تلقائيا فى صفحة "جدول الرموز".

المفاتيح المستخدمة فى صفحة الرموز.



١- Toggles POU comments:

- بالضغط على هذا المفتاح تظهر التعليقات الخاصة بصفحات البرمجة.

PROGRAM COMMENTS

٢- Toggles NETWORK comments:

- بالضغط على هذا المفتاح تظهر التعليقات الخاصة بفروع البرمجة.

Network Comment

٣- Toggles symbol information table:

- بالضغط على هذا المفتاح تظهر الرموز و التعليقات الخاصة بكل عنوان فى جدول.

Symbol	Address	Comment

٤- Apply all symbols in project:

- فى حالة كتابة الرموز بعد الانتهاء من البرمجة يتم الضغط على هذا المفتاح لكى يقوم بإظهار الرموز بجوار العناوين.

٥- Create table undefined symbols:

- فى حاله عمل برنامج باستخدام الرموز ودون كتابة أى عناوين يتم الضغط على هذا المفتاح لكى يقوم البرنامج تلقائيا بإظهار الرموز التى ليس لها عناوين فى جدول لكي يقوم المبرمج بكتابة العناوين.

توجد طريقتان لاستخدام جدول الرموز فى البرمجة:

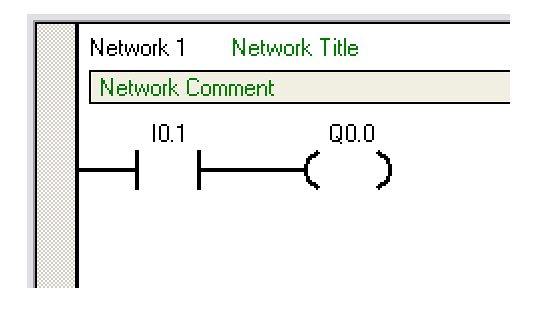
١ – البرمجة باستخدام العناوين ثم إعطاء رمز لكل عنوان.

٢ – البرمجة باستخدام الرموز ثم إعطاء عنوان لكل رمز.

الطريقة الأولى:

البرمجة باستخدام العناوين أولاً ثم إعطاء رمز لكل عنوان.

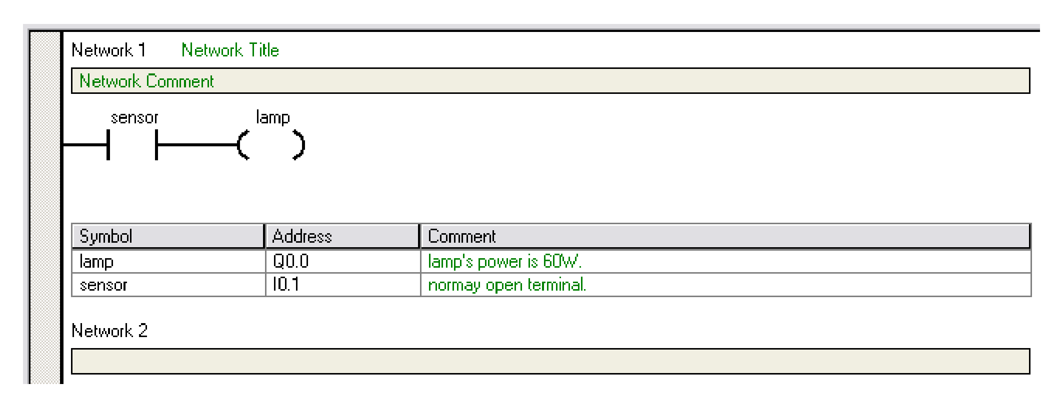
١ - رسم البرنامج باستخدام العناوين.



٢ - كتابة رمز لكل عنوان.

			Symbol	Address	Comment
1			sensor	I0.1	normay open terminal.
2			lamp	Q0.0	lamp's power is 60W.
3					
4					
5					

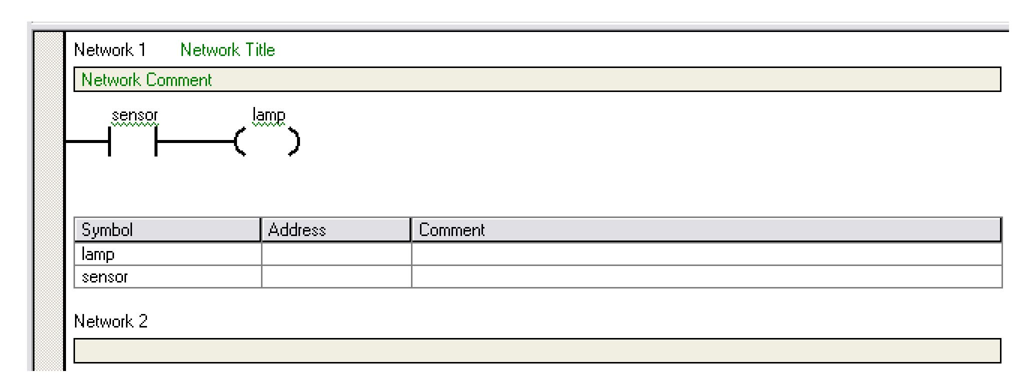
٣ -الضغط على Apply all symbols in project لتفعيل التغيير.



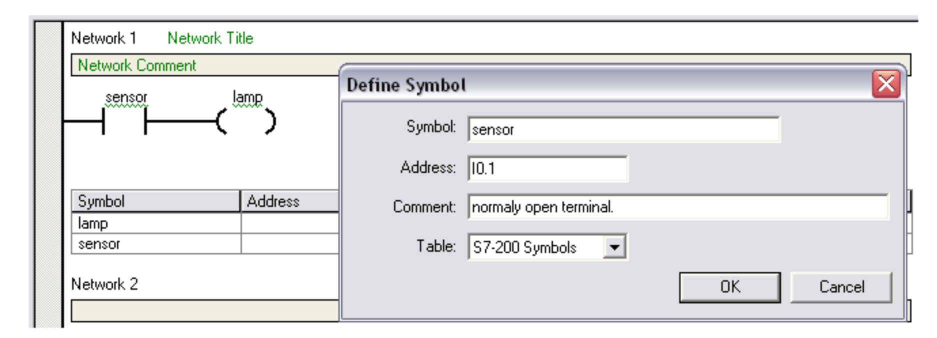
الطريقة الثانية:

البرمجة باستخدام الرموز أولاً ثم إعطاء عنوان لكل رمز.

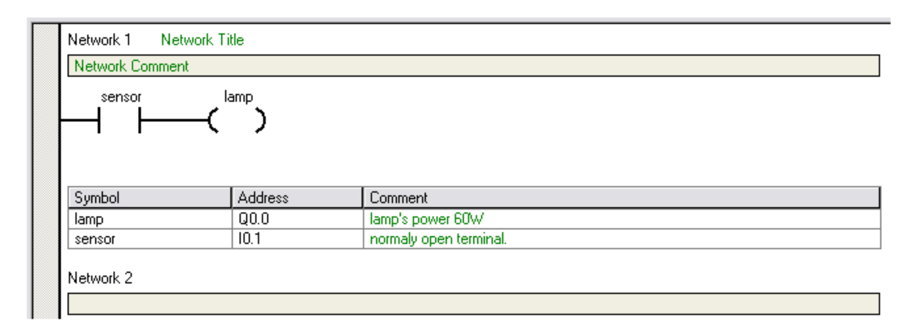
١ - رسم البرنامج باستخدام الرموز.

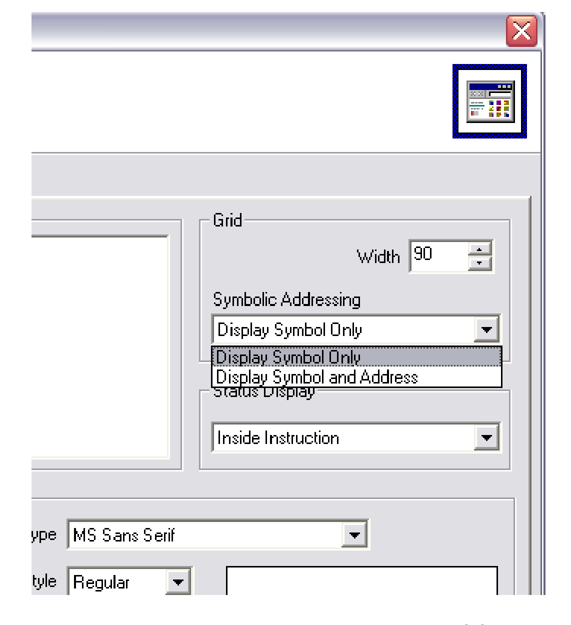


٢ - كتابة عنوان لكل رمز بالضغط مرتين على كل رمز.



٣ -بعد الانتهاء من الرموز يصبح البرنامج بهذا الشكل.





خصائص:

لظهور الرموز و العناوين معاً أو لظهور الرموز فقط بدون العناوين يتم الضغط على tools ثم options ثم الاختيار بين النوعين:

١ – عرض الرموز فقط

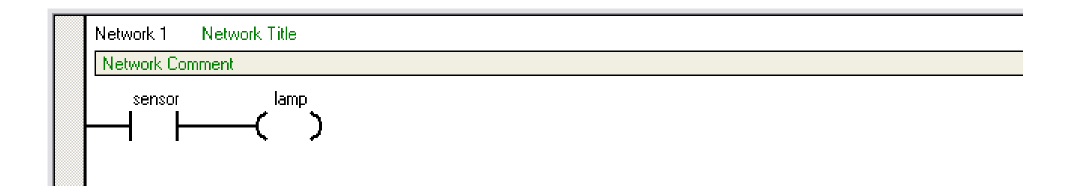
display symbol only

٢ – عرض الرموز والعناوين

display symbol and address

بعد الانتهاء من وضع الرموز الخاصة بكل عنوان يمكن إظهار الرموز بالطريقتين التاليتين:

- **الطريقة الأولى**: (الرموز فقط)



- **الطريقة الثانية**: (الرموز و العناوين معاً)

Network 1 Network Title
Network Comment
sensor:I0.1 lamp:Q0.0

ملاحظة:

- ويتم تطبيق الطريقة الأولى باختيار Display Symbol Only ولكن لا يفضل هذه الطريقة بسبب التخبط الذى ينتج بسببها.
- ويتم تطبيق الطريقة الثانية باختيار Display Symbol And Address وهى الطريقة الأفضل فى عرض الرموز وبالأخص فى البرامج التى تحتوى على عدد كبير من أفرع البرمجة network.
- يرجى عدم كتابة رموز تتكون من أحرف كثيرة لتجنب التخبط فى حين تطبيق الرموز على البرنامج بالفعل

ومن ضن الأعمال الهامة لصفحة "جدول الرموز" أنها تحتوى على جدول فيه مئات العناوين الخاصة بالعمليات الأكثر استخداماً بل ويوجد أيضا تعليق خاص بكل عنوان كما هى موضحاً بالصورة التالية و سوف يتم شرح بعض الأمثلة منها بالغة العربية في الفصل التاسع بالتفصيل

			Symbol	Address	Comment
1			Always_On	SM0.0	Always ON
2			First_Scan_On	SM0.1	ON for the first scan cycle only
3			Retentive_Lost	SM0.2	ON for one scan cycle, if retentive data is lost
4			RUN_Power_Up	SM0.3	ON for 1 scan cycle when RUN mode is entered from a power-up condition
5			Clock_60s	SM0.4	Clock pulse that is ON for 30 s, OFF for 30 s, for a duty cycle time of 1 min.
6			Clock_1s	SM0.5	Clock pulse that is ON for 0.5 s, OFF for 0.5 s, for a duty cycle time of 1 s.
7			Clock_Scan	SM0.6	Scan cycle clock which is ON for one cycle and OFF for the next cycle
8			Mode_Switch	SM0.7	Indicates the current position of the mode switch: 0 = TERM, 1 = RUN
9			Result_0	SM1.0	Set to 1 by the execution of certain instructions when the operation result = 0
10			Overflow_Illegal	SM1.1	Set to 1 by exec. of certain instructions on overflow or illegal numeric value.
11			Neg_Result	SM1.2	Set to 1 when a math operation produces a negative result
12			Divide_By_0	SM1.3	Set to 1 when an attempt is made to divide by zero
13			Table_Overflow	SM1.4	Set to 1 when the Add to Table instruction attempts to overfill the table
14			Table_Empty	SM1.5	Set to 1 when a LIFO or FIFO instruction attempts to read from an empty table
15			Not_BCD	SM1.6	Set to 1 when an attempt is made to convert a non-BCD value to a binary value
16			Not_Hex	SM1.7	Set to 1 when an ASCII value cannot be converted to a valid hexadecimal value
17			Parity_Err	SM3.0	Set to 1 if a parity error is detected in a char received by Port 0 or Port 1
18			Comm_Int_Ovr	SM4.0	Set to 1 if the communication interrupt queue overflows (interrupt routine only)
19			Input_Int_Ovr	SM4.1	Set to 1 if the input interrupt queue overflows (interrupt routine only)
20			Timed_Int_Ovr	SM4.2	Set to 1 if the timed interrupt queue overflows (interrupt routine only)

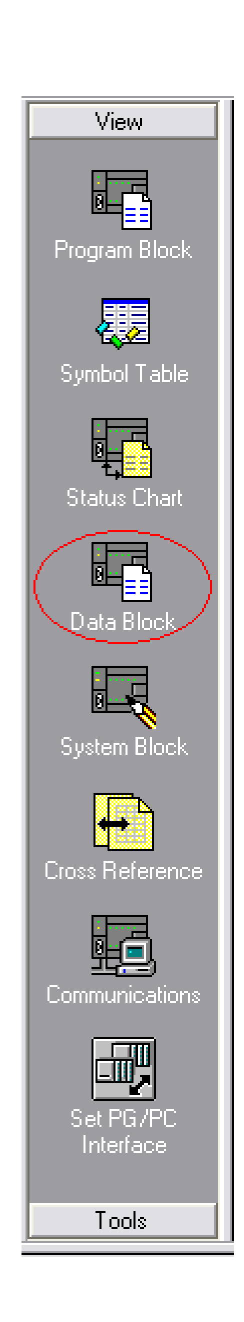
S7-200 Symbols | USER2 | POU Symbols

الباب الرابع

صفحة البيانات

- شرح صفحة البيـــــــــــــــــــــــــــانات.
- محتويات صفحة البيــــــــــــــــــــــانات.
- المفاتيح المستخدمة فى صفخة البيـــــــــــــــانات.
- الأخـــــــــــــــــطاء الممكن التعرض لها.
- التعليقات الخاصة بصفحة البيـــــــــــــــــانات.
- كيفية تخزين قيم مسبقة على متغيـــــرات بحجم Bit.
- كيفية تخزين قيم مسبقة على متغيـــرات بحجم Byte.
- كيفية تخزين قيم مسبقة على متغيـــرات بحجم Word.
- كيفية تخزين قيم مسبقة على متغيرات بحجم Dword.
- تمـــــــــــــــــــــارين عملية للتوضيح.

صفحة البيانات Data Block :



تستخدم صفحة " البيانات" الـ Data Block كصفحة للتعليقات أو كصفحة لتحديد قيم مسبقة خاصة بالمتغيرات بجميع أنواعها سواء المستخدمة فى البرنامج أو غير المستخدمة فى البرنامج وهذا لأن فى جميع الحالات تكون القيمة الأولية للمتغيرات هى صفر وهذا قد يسبب بعض المشاكل وهذا يحدث أذا تم استخدام المتغيرات بالتحديد مع المؤقتات الزمنية, مع العدادات أو حتى مع مفاتيح المقارنة كما سوف يتم التوضيح فى التمارين العملية فى نفس الفصل.

استخدامات صفحة " البيانات":

- استخدام صفحة البيانات للتعليقات:

يمكن كتابة التعليقات قبل أو بعد تحميل البرنامج و هذا لأن التعليقات لا تعنى أى شيء بالنسبة لجهاز الـ PLC, فمثلاً لن يؤثر على البرنامج أذا كتبت كتعليق أن ثمن المفتاح هو عشرة جنيهات أو أن لون الموتور أزرق.

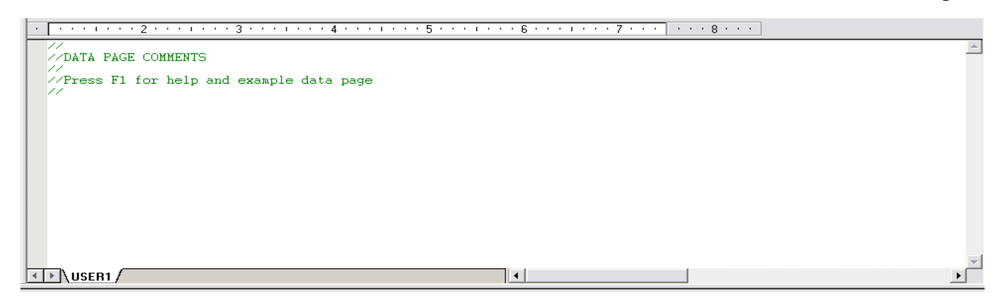
– استخدام صفحة البيانات للمتغيرات:

يجب كتابة القيم الخاصة بالمتغيرات قبل تحميل البرنامج و ليس بعد و هذا لأن تلك القيم تعنى الكثير لجهاز ال PLC لأنها ستؤثر على البرنامج, فمثلاً أذا كتبت أن قيمة المتغير VW0 هى ٤٠٠٠ فمثلاً أذا كان هذا المتغير مستخدم مع مؤقت زمنى T32 فهذا يعنى أن المؤقت الزمنى سوف يغير النقط الخاصة به بعد أربع ثوانى من التشغيل.

ملاحظة:

- فى حالة عدم كتابة أى قيمة مسبقة للمتغيرات فهذا يعنى أن القيمة الحالية للمتغيرات ستكون صفر و قد يتسبب هذا فى بعض المشاكل.
- يمكن تغير قيمة أى متغير بواسطة صفحة "جدول الحالات" حتى ولو كان لهذا المتغير قيمة مسبقة, ويتم هذا بواسطة أمر write all أنظر صفحة ٤٢.
- لكتابة بعض التعليقات يجب إن تكتب فى البداية "//" ثم يكتب التعليق المراد كتابته.
- لكتابة بعض المتغيرات يجب إن لا تكتب فى البداية "//" بل يكتب المتغير المراد كتابته دون أى مقدمات.
- فى حالة كتابة أى متغيرات بطريقة خطاء أو كتابة أى تعليق دون "//" تظهر تلقائيا علامة "X" لكى تشير إلى أنه قد تم كتابة شىء غير صحيح.
- الطريقة الأفضل لتغيير محتويات المتغيرات بواسطة البرنامج تتم عن طريق عمليات النقل, أنظر صفحة ٢٠٩ فى الجزء الأول من الكتاب.

شكل صفحة البيانات:



المفاتيح المستخدمة فى صفحة البيانات.



- Compile all: بالضغط على هذا المفتاح يقوم البرنامج بمراجعة صفحة البيانات لإظهار الأخطاء أذا وجدت.

- Download: بالضغط على هذا المفتاح يقوم بتحميل البرنامج كما سبق و تحميل صفحة البيانات أيضاً.

مثال عملى باستخدام صفحة "جدول المرجع":

محرك يعمل يميناً لوقت ثم يقف لوقت ثم يعمل تلقائياً لليسار لوقت أخر ثم يقف لوقت و هكذا بشرط أن يكون هناك أمكانية لتغير الزمن.

عدد الدخل	نوع الدخل	أسم الدخل
١	n.c.	I0.0/S1
٢	n.o.	I0.1/S2
عدد المتغيرات	**نوع المتغيرات**	**أسم المتغيرات**
١	<=I	T200/VW0

٢	>=I	T200/ VW2
٣	<=I	T200/VW4
٤	<=I	T200/ VW6
عدد المتغيرات	**نوع المتغيرات**	**أسم المتغيرات**
١	Word	VW0
٢	Word	VW2
٣	Word	VW4
٤	Word	VW6
عدد الريليهات	**نوع الريليهات**	**أسم الريليهات**
١	Bit	M0.0
عدد المؤقتات الزمنية	**نوع المؤقتات الزمنية**	**أسم المؤقتات الزمنية**
١	TON	T200
عدد الخرج	**نوع الخرج**	**أسم الخرج**
١	كونتكتور	Q0.1/K1M
٢	كونتكتور	Q0.2/K2M

الشرح:

Network1: بالضغط على I0.1 يعمل الريليه M0.0 بشرط أن يكون I0.0 مغلق.

Network2: يعمل المؤقت الزمني T96 لزمن متغير بقيمة مسبقة ثانيتان ثم يقف ليبدأ من جديد.

Network3: يعمل المحرك يميناً لزمن متغير بقيمة مسبقة ثانيتان ثم يقف لزمن متغير بقيمة مسبقة ثانيتان أيضاً.

Network4: يعمل المحرك يساراً لزمن متغير بقيمة مسبقة ثانيتان ثم يقف لزمن متغير بقيمة مسبقة ثانيتان أيضاً.

البرنامج:

Network 1 Network Title

Network Comment

I0.1 I0.0 M0.0

M0.0

Network 2

M0.0 T200 T200

<=I IN TON

VW6

12 PT 100 ms

Network 3

M0.0 T200 Q0.1

<=I

VW0

Network 4

M0.0 T200 T200 Q0.2

>=I <=I

VW2 VW4

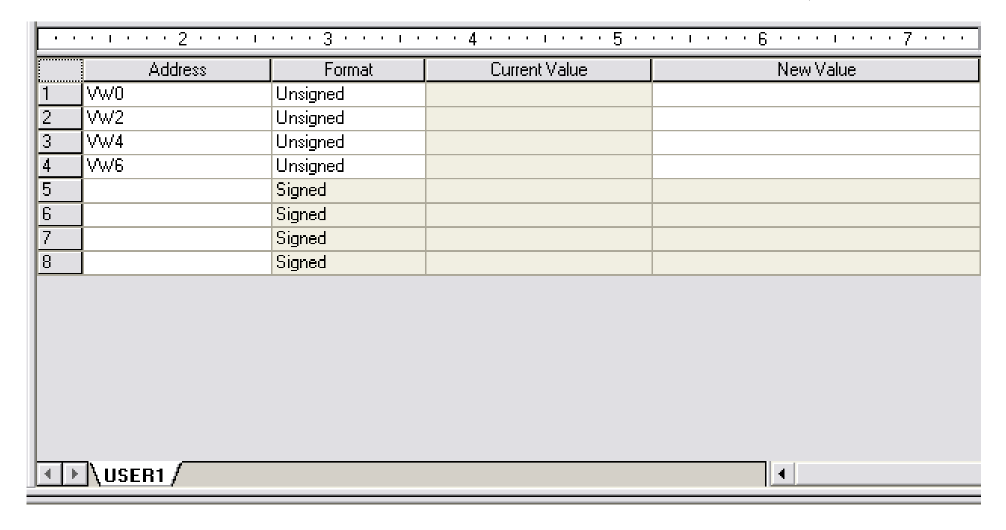
تم تحديد قيم المتغيرات بواسطة صفحة البيانات كما سبق وشرحنا.

```
//
//DATA PAGE COMMENTS
//
//Press F1 for help and example data page
//


VW0    20                          // VARIABLE MEMORY VW0(SIZE WORD).
VW2    40                          // VARIABLE MEMORY VW2(SIZE WORD).
VW4    60                          // VARIABLE MEMORY VW4(SIZE WORD).
VW6    80                          // VARIABLE MEMORY VW6(SIZE WORD).
```

USER1

بل يمكن أيضاً تغير قيم المتغيرات مرة أخرى بواسطة صفحة "جدول الحالات" بواسطة أمر write all.



	Address	Format	Current Value	New Value
1	VW0	Unsigned		
2	VW2	Unsigned		
3	VW4	Unsigned		
4	VW6	Unsigned		
5		Signed		
6		Signed		
7		Signed		
8		Signed		

USER1

بالضغط على مفتاح Compile all يقوم البرنامج بمراجعة صفحة البيانات لإظهار الأخطاء أذا وجدت.

```
Compiling Program Block...
MAIN (OB1)
SBR_0 (SBR0)
INT_0 (INT0)
Block Size = 64 (bytes), 0 errors

Compiling Data Block...
Block Size = 8 (bytes), 0 errors

Compiling System Block...
Compiled Block with 0 errors, 0 warnings

Total Errors: 0
```

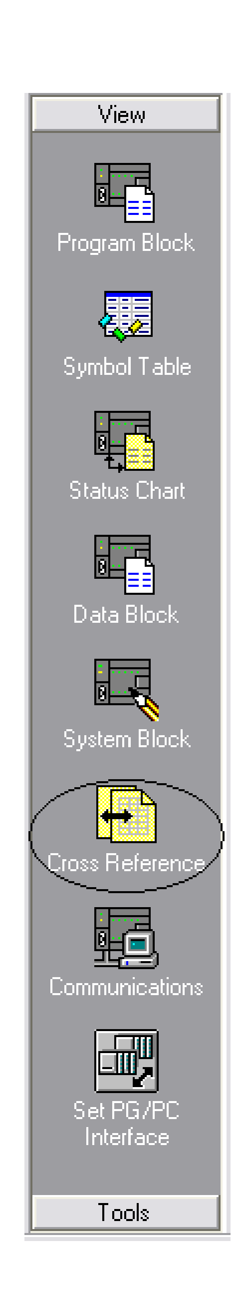
Ready

الباب الخامس

جدول المرجع

- شرح جدول المـــــــــــــــرجع.
- محتويات جدول المـــــــــــــرجع.
- صـــــــــــفحة Cross Reference.
- صـــــــــــــــفحة Byte Usage.
- صـــــــــــــــفحة Bit Usage.
- المفاتيح المستخدمة فى جدول المرجع.
- الأخـــــــــــطاء الممكن التعرض لها.
- تمـــــــــــــــــارين عملية للتوضيح.

جدول المرجع Cross Reference :



تستخدم صفحة "جدول المرجع" الـ Cross Reference لكى تستخدم كمرجع لكل العناوين المستخدمة فى البرنامج بجميع أنوعها. حيث تنقسم هذه الصفحة إلى ثلاث صفحات فرعية:

الصفحة الفرعية الأولى تستخدم لعرض جميع العناوين مثل: المداخل, المخارج, الريليهات, المتغيرات, العدادات, المؤقتات, الخ.....

الصفحة الفرعية الثانية تستخدم لعرض بعض العناوين مثل: المتغيرات, العدادات و المؤقتات.

الصفحة الفرعية الثالثة تستخدم لعرض عناوين بحجم bit مثل: المداخل, المخارج و الريليهات.

طرق استخدام صفحة "جدول المرجع":

– يتم عمل البرنامج أولاً ثم تحميل البرنامج ثم فتح صفحة "جدول المرجع".

– يتم عمل البرنامج أولاً ثم الضغط على compile ثم فتح صفحة "جدول المرجع".

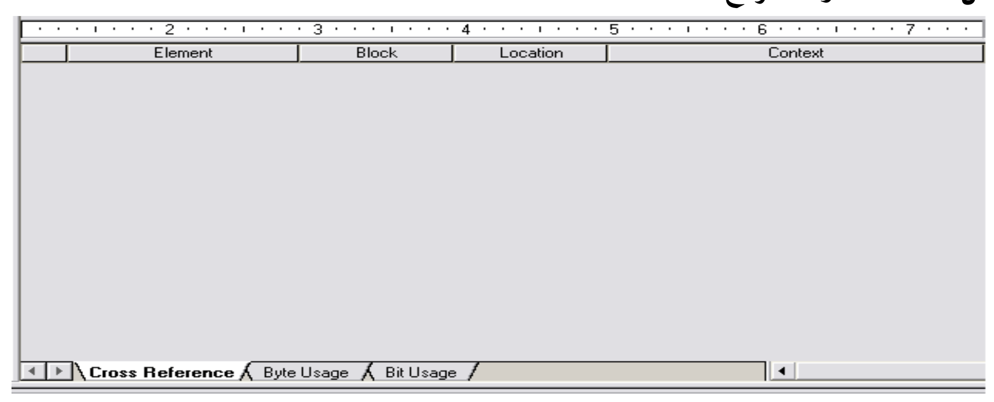
ملاحظة:

فى حاله رسم البرنامج ثم فتح صفحة "جدول المرجع" مباشر دون تحميل البرنامج أو الضغط على compile تظهر هذه الرسالة:

A compile must be performed to display cross reference.

أى أنه يجب الضغط على compile لإظهار صفحة "جدول المرجع".

شكل صفحة "جدول المرجع":



الصفحة الفرعية الأولى Cross Reference **تحتوى على:**

Element – Block – Location – Contexet

١–Element: حيث تظهر كل العناوين التى استخدمت فى البرنامج و أذا قد استخدم العنوان أكثر من مرة فأنه يظهر بنفس عدد المرات التى تكرر بها فى البرنامج.

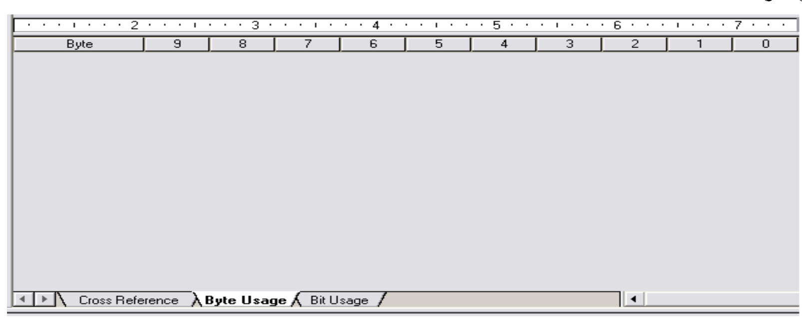
٢–Block: حيث يظهر لك فى أى صفحة برمجة (MAIN أو SBR أو INT) يوجد هذا العنوان.

٣–Location: حيث يشير فى أى فرع من فروع (network 1 أو network 2 أو .....) البرنامج قد استخدم هذا العنوان.

٤-Context: حيث يضيف بعض التوضيحات الخاصة بالعنوان, فمثلاً أذا كان العنوان مفتاح فأنه يوضح أذا كان مفتوح أو مغلق أو مفتاح مقارنة, أو أذا كان العنوان هو لمؤقت زمنى فأنه يوضح أذا كان نوعه TON أو TOF أو TONR, أو أذا كان العنوان هو لعداد فأنه يوضح أذا كان نوعه CTU أو CTD أو CTUD وهكذا....

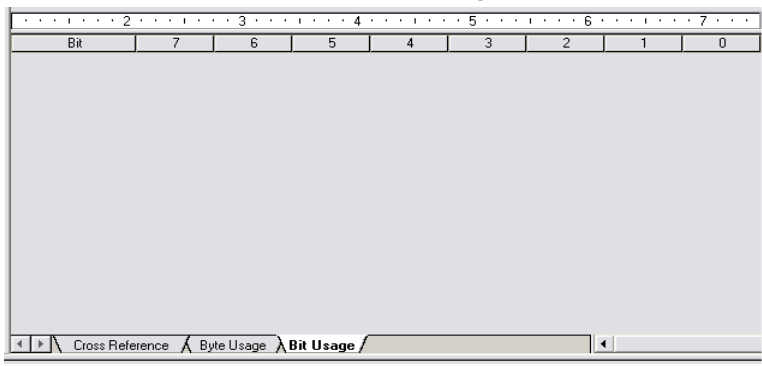
الصفحة الفرعية الثانية Byte Usage:

تستخدم لإظهار أسم العدادات و المؤقتات الزمنية و يشير أيضاً إلى مجموعة الـ byte المستخدمة سواء للمتغيرات أو للريليهات



الصفحة الفرعية الثالثة Bit Usage:

حيث يستخدم لإظهار أسم المداخل, المخارج أو الريليهات ولكن كل عنوان على حدا



حيث يستخدم لإظهار أسم العناوين التي هي مثل: الدخل, الخرج, الريليه أو المتغير. أى أنه يشير إلى عناوين بحجم bit.

المفاتيح المستخدمة فى صفحة الرموز.



Compile: بالضغط على هذا المفتاح تصبح صفحة "جدول المرجع" متاحة.

Compile all بالضغط على هذا المفتاح أيضاً تصبح صفحة "جدول المرجع" متاحة.

Download: بالضغط على هذا المفتاح يقوم بتحميل البرنامج كما سبق و شرحنا و تصبح صفحة "جدول المرجع" متاحة.

مثال عملى باستخدام صفحة "جدول المرجع":

محرك يعمل يميناً لوقت ثم يقف لوقت ثم يعمل تلقائياً لليسار لوقت أخر ثم يقف لوقت و هكذا.

عدد الدخل	نوع الدخل	أسم الدخل
١	n.c.	I0.0/S1
٢	n.o.	I0.1/S2
عدد المتغيرات	**نوع المتغيرات**	**أسم المتغيرات**
١	>=I	T96
٢	<=I	T96
٣	<=I	T96

٤	<>I	T96
عدد الريلهات	**نوع الريلهات**	**أسم الريلهات**
١	Bit	M0.0
عدد المؤقتات	**نوع المؤقتات**	**أسم المؤقتات**
١	TON	T96
عدد الخرج	**نوع الخرج**	**أسم الخرج**
١	كونتكتور	Q0.1/K1M
٢	كونتكتور	Q0.2/K2M

البرنامج:

Network 1 Network Title
Network Comment
I0.1 I0.0 M0.0
M0.0

Network 2
M0.0 T96 <>I 8000 T96 IN TON 2 PT 1 ms

Network 3
M0.0 T96 <=I 2000 Q0.1

Network 4
M0.0 T96 >=I 4000 T96 <=I 6000 Q0.2

الشرح:

Network1:

بالضغط على I0.1 يعمل الريليه M0.0 بشرط أن يكون I0.0 مغلق.

Network2:

يعمل المؤقت الزمنى T96 لمدة ثمان ثواني ثم يقف ليبدأ من جديد.

Network3:

يعمل المحرك يميناً لمدة ثانيتان (من صفر إلى أثنين) ثم يقف لمدة ثانيتان أخريتان (من أثنين إلى أربعة).

Network4:

يعمل المحرك يساراً لمدة ثانيتان (من أربعة إلى ستة) ثم يقف لمدة ثانيتان أخريتان (من ستة إلى ثمانية).

صفحة Cross Reference **الخاصة بهذا البرنامج:**

<table>
<tr><th></th><th>Element</th><th>Block</th><th>Location</th><th>Context</th></tr>
<tr><td>1</td><td>I0.0</td><td>MAIN (OB1)</td><td>Network 1</td><td>-| |-</td></tr>
<tr><td>2</td><td>I0.1</td><td>MAIN (OB1)</td><td>Network 1</td><td>-| |-</td></tr>
<tr><td>3</td><td>Q0.1</td><td>MAIN (OB1)</td><td>Network 3</td><td>-( )</td></tr>
<tr><td>4</td><td>Q0.2</td><td>MAIN (OB1)</td><td>Network 4</td><td>-( )</td></tr>
<tr><td>5</td><td>M0.0</td><td>MAIN (OB1)</td><td>Network 1</td><td>-( )</td></tr>
<tr><td>6</td><td>M0.0</td><td>MAIN (OB1)</td><td>Network 1</td><td>-| |-</td></tr>
<tr><td>7</td><td>M0.0</td><td>MAIN (OB1)</td><td>Network 2</td><td>-| |-</td></tr>
<tr><td>8</td><td>M0.0</td><td>MAIN (OB1)</td><td>Network 3</td><td>-| |-</td></tr>
<tr><td>9</td><td>M0.0</td><td>MAIN (OB1)</td><td>Network 4</td><td>-| |-</td></tr>
<tr><td>10</td><td>T96</td><td>MAIN (OB1)</td><td>Network 2</td><td>-|<>I|-</td></tr>
<tr><td>11</td><td>T96</td><td>MAIN (OB1)</td><td>Network 2</td><td>TON</td></tr>
<tr><td>12</td><td>T96</td><td>MAIN (OB1)</td><td>Network 3</td><td>-|<=I|-</td></tr>
<tr><td>13</td><td>T96</td><td>MAIN (OB1)</td><td>Network 4</td><td>-|>=I|-</td></tr>
<tr><td>14</td><td>T96</td><td>MAIN (OB1)</td><td>Network 4</td><td>-|<=I|-</td></tr>
</table>

Cross Reference | Byte Usage | Bit Usage

صفحة Byte Usage الخاصة بهذا البرنامج:

Byte	9	8	7	6	5	4	3	2	1	0
MB0										b
T0										
T10										
T20										
T30										
T40										
T50										
T60										
T70										
T80										
T90				X						

Cross Reference | **Byte Usage** | Bit Usage

ملاحظة:

⇐ تحتوى هذه الصفحة على مسميات المؤقتات الزمنية المستخدمة فى البرنامج, بحيث يكتب الاسم بطريقة خاصة فتكتب مسميات المؤقتات بداياً من صفر إلى أن يصل إلى أسم المؤقت المستخدم فعلاً و تشير أيضاً إلى أنه أستخدم الريليه الأول.

⇐ فى حالة استخدام ذاكرة "متغيرات" بحجم word مثلاً فأنه يتم الإشارة إلى الـ bytes المكونة لهذا الـ word فمثلاً أذا كان المتغير المستخدم فى البرنامج هو VW4 فأنة يشير إلى هذه الذاكرة داخل صفحة "جدول المرجع" بواسطة كتابة حرف الـ W أمام الـ byte4 والـ byte5 ليشير إلى أنة تم استخدام الـ byte4 والـ byte5 ولكن عن طريق الـ word4.

⟸ فى حالة استخدام ذاكرة "متغيرات" بحجم Dword مثلاً فأنه يتم الإشارة إلى الـ bytes المكونة لهذا الـ Dword فمثلاً أذا كان المتغير المستخدم فى البرنامج هو VD0 فأنة يشير إلى هذه الذاكرة داخل صفحة "جدول المرجع" بواسطة كتابة حرف الـ D أمام الـ byte0 والـ byte1 والـ byte2 والـ byte3 ليشير إلى أنة تم استخدام كل هذه الـ bytes ولكن عن طريق الـ Dword0.

صفحة Bit Usage الخاصة بهذا البرنامج:

Bit	7	6	5	4	3	2	1	0
I0.0							b	b
Q0.0						b	b	
M0.0								b

Cross Reference | Byte Usage | **Bit Usage**

هذه الصفحة تشير إلى كل العناوين التى استخدمت فى البرنامج بشرط أن تكون بحجم bit, مثل:

I0.0, I0.1, Q0.1, Q0.2 و M0.0

الباب السادس

البرامج الفرعية

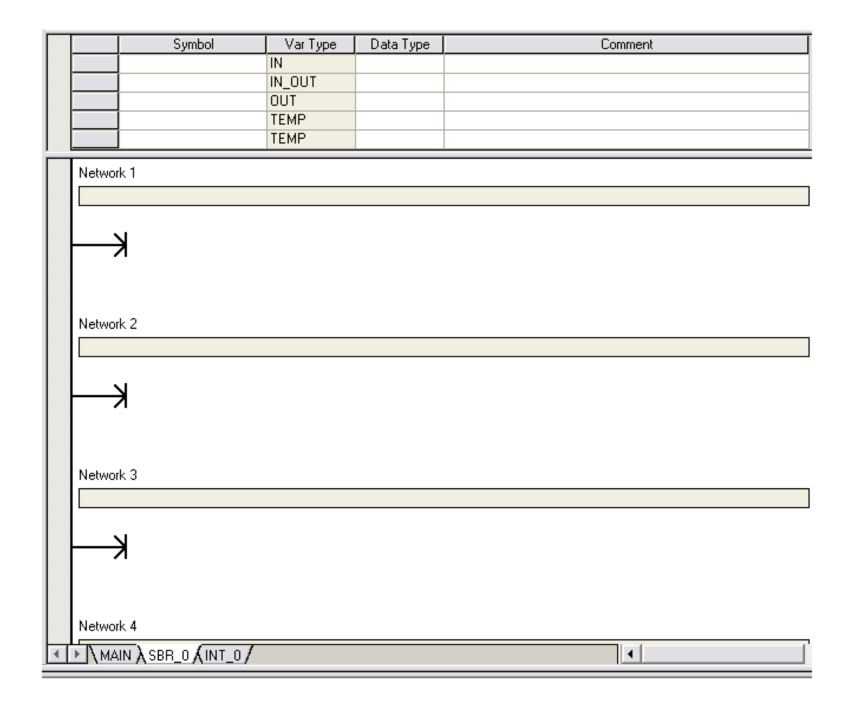
- شرح البرامـــــــــــــــــــــــــــــــج الفرعية.
- محتويات البرامـــــــــــــــــــــــــــــج الفرعية.
- الجدول المستخدم فى البرامـــــــــــــــج الفرعية.
- الأخطاء الممكن حدوثها فى جدول البرامج الفرعية.
- التعليقات الخاصة بصفحة البرامـــــــــــج الفرعية.
- جدول الرموز الخاص بصفحات البرمجـــــــــــــة.

صفحة البرامج الفرعية Subroutine :

تستخدم طريقة "البرامج الفرعية" الـ Subroutine فى:

١- البرامج الكبيرة التى تحتوى على أجزاء متكررة.

٢- لتقسيم البرنامج الكبير إلى أجزاء صغيرة لسهولة حل الأعطال.



الحالة الأولى:

- فى حاله وجود نظام الإنذارات الخاص بكل مكينة فأن الإنذارات تعمل بنفس الطريقة حتى مع اختلاف الماكينات .
- فى حالة وجود لمبات أشارة تضىء مع كل محرك فأن لمبات تعمل بنفس الطريقة حتى مع اختلاف المحركات.

- فى حاله وجود فلاشر يضاء حسب شروط خاصة فأن الفلاشر يعمل بنفس الطريقة حتى مع اختلاف الشروط .

ملاحظة:

أى أنة مثلاً بدلاً من تكرار أى جزء عشرة مرات فأنة يرسم مرة واحدة فقط ولكن يتم أعطاء هذا الجزء عشرة عناوين مختلفة كما لو كان مرسوم عشرة مرات فعلاً.

الحالة الثانية:

- فى حاله وجود برنامج لمجموعة ماكينات مختلفة فأنة يفضل استخدام صفحات مختلفة لكل مكينة حيث أنه بتقسيم البرنامج يسهل على المبرمج التوصل لسبب الأعطال بسهولة وفى وقت قصير.

طريقة استخدام صفحة "البرامج الفرعية":

- **أولاً** يتم رسم الجزء الذى كان من المفروض أن يتكرر داخل صفحة " البرامج الفرعية " ولكن مرة واحدة فقط ودون أعطاء أى عناوين محددة.

	Symbol	Var Type	Data Type	Comment
		IN		
		IN_OUT		
		OUT		
		TEMP		
		TEMP		

Network 1 Network Title

Network Comment

??.? ??.? ??.?

- **ثانياً** يتم تحديد كل المسميات التى سوف تستخدم فى البرنامج الفرعى داخل الجدول الخاص بالبرامج الفرعية مع تحديد أذا كان هذا الاسم هو لدخل أو لخرج أو ....

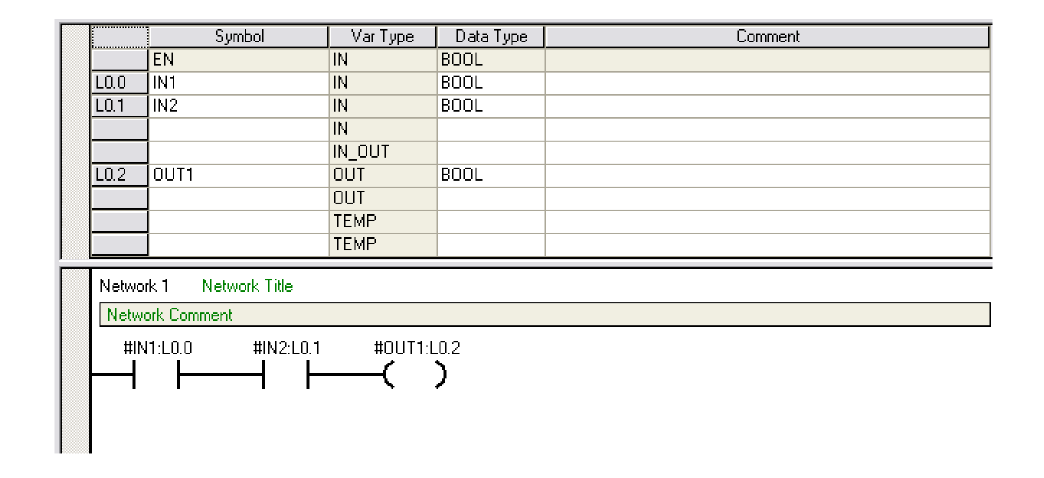
	Symbol	Var Type	Data Type	Comment
	EN	IN	BOOL	
L0.0	IN1	IN	BOOL	
L0.1	IN2	IN	BOOL	
		IN		
		IN_OUT		
L0.2	OUT1	OUT	BOOL	
		OUT		
		TEMP		
		TEMP		

الجدول الخاص بالبرامج الفرعية.

جدول الرموز Var Type الخاص بصفحة البرامج الفرعية يحتوى على:

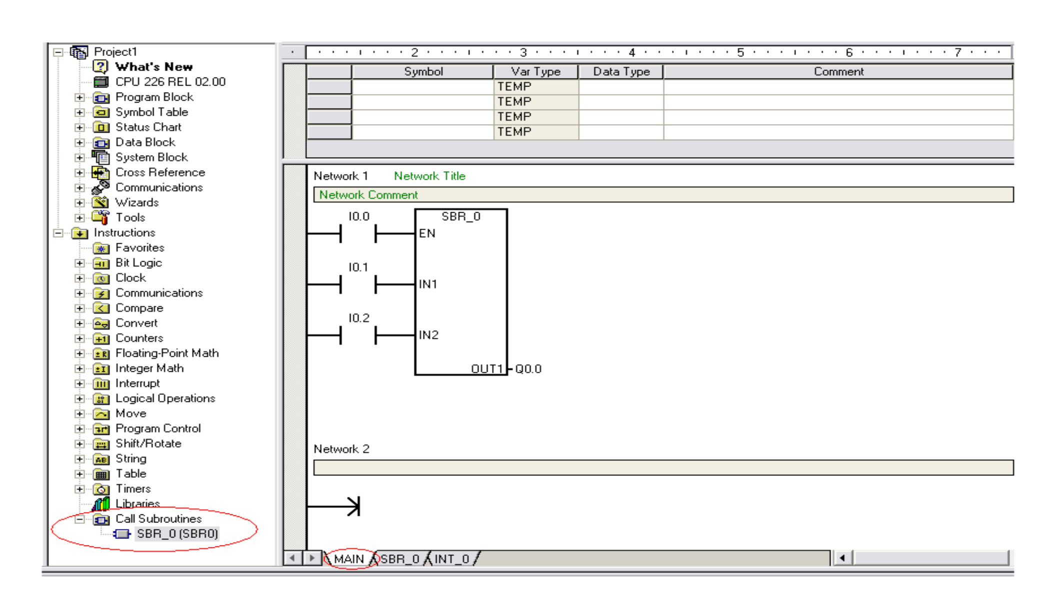
- IN: أى أنه يستخدم فى البرامج الفرعية كدخل.
- OUT: أى أنه يستخدم فى البرامج الفرعية كخرج.
- IN_OUT: أى أنه يستخدم فى البرامج الفرعية كدخل و خرج مثل الريله.
- TEMP: أى أنه يستخدم فى البرامج الفرعية مثل المتغيرات
- BOOL: تستخدم مع عناوين بحجم bit.
- BYTE: تستخدم مع عناوين بحجم byte.
- WORD: تستخدم مع عناوين بحجم word(unsigned).
- INT: : تستخدم مع عناوين بحجم word(signed).
- DWORD: تستخدم مع عناوين بحجم Dword(unsigned).
- DINT: تستخدم مع عناوين بحجم Dword(signed).
- REAL: تستخدم مع عناوين بحجم Dword(real).
- STRING: تستخدم مع عناوين بحجم byte.

- **ثالثاً** يتم كتابة الأسم الذى تم تحديدة فى البرنامج بدل العنوان و الهدف من هذا أنة أذا كتبنا عنوان فهذا يعنى أنة تم اختيار العنوان إلى الأبد ولكن فى حاله كتابة أى رمز فأنة يمكن بكل سهوله فى ما بعد أعطاء عناوين مختلفة لنفس الرمز.



	Symbol	Var Type	Data Type	Comment
	EN	IN	BOOL	
L0.0	IN1	IN	BOOL	
L0.1	IN2	IN	BOOL	
		IN		
		IN_OUT		
L0.2	OUT1	OUT	BOOL	
		OUT		
		TEMP		
		TEMP		

- **رابعاً** يتم الاتصال بصفحة "البرامج الفرعية" عن طريق أمر CALL_SUB الذى يوجد فى صفحة "البرنامج الرئيسية" مع تحديد العنوان الذى سيحل محل الرمز.



الأخطاء الممكن حدوثها فى جدول البرامج الفرعية:

١– ممنوع كتابة أى رمز يكون بالصدفة أسم من أسماء العماليات المستخدمة.

- أى أنه لا يمكن كتابة كلمة ("stop", "S", "R", "JMP", "LBL", "END", "ENI", "TON", "TOF", "TONR", "CTU", "CTD", "CTUD") حيث أن كل هذه الكلمات السابقة تمثل عمليات تستخدم فى جهاز ال PLC.

٢– ممنوع كتابة أى رمز مكون من كلمتين و يكون بينهم مسافة.

- أى أنه لا يمكن كتابة كلمة ("first button", "second button ", "motor right", "motor left") حيث أن المسميات المكتوبة هى مكونة من كلمتين ولكن لحل هذه المشكلة يتم وضع أى رمز أو علامة بين الكلمة الأولى والكلمة الأخرى.

٣– ممنوع تكرار نفس الرمز مع عناوين مختلفة.

- أى أنه لا يمكن كتابة أى رمز أكثر من مرة حتى ولو كان مع نفس العنوان حيث أن فى هذه الحالة يصعب على ال PLC وضع رمزين مع نفس العنوان فيترك العنوان بدون رمز .

٤– اختلاف بين الرموز المحددة فى الجدول و المستخدمة فى البرنامج الفرعى.

- أى أنه لا يمكن كتابة أى أسم فى الجدول بحروف كبيرة بينما يكتب فى البرنامج الفرعى بحروف صغيرة أو حتى أن يعرف العنوان فى الجدول بأسم IN1 بينما يستخدم فى البرنامج الفرعى بأسم in1 لأن بالنسبة لوحدة البرمجة ال PLC يوجد فرق كبير بين الأسمين.

المفاتيح المستخدمة فى صفحة البرامج الفرعية



١- Program Status:

بالضغط على هذا المفتاح تظهر الحالة الخاصة بكل عنوان.

٢- Compile all:

بالضغط على هذا المفتاح تظهر عدد الأعطال الخاصة بالبرنامج.

مثال عملى:

محركين يعمل كل محرك فى اتجاهين بحيث أن مع كل أتجاة تعمل لمبتان كفلاشر بشرط أن اللمبات الخاصة بالأتجاة اليمين مختلفة عن اللمبات الخاصة بالاتجاه اليسار و اللمبات الخاصة بالمحرك الأول مختلفة عن المحرك الثانى.

المداخل

عدد الدخل	نوع الدخل	أسم الدخل
١	n.c.	I0.0/S1
٢	n.o.	I0.1/S2
٣	n.o.	I0.2/S3
٤	n.o.	I0.3/S4
٥	n.o.	I0.4/S5
٦	n.o.	I1.1/S6
٧	n.o.	I1.2/S7
٨	n.o.	I1.3/S8
٩	n.o.	I1.4/S9

المؤقتات الزمنية

عدد المؤقتات الزمنية	نوع المؤقتات الزمنية	أسم المؤقتات الزمنية
١	TON	T32
٢	TON	T96

المخارج

عدد الخرج	نوع الخرج	أسم الخرج
١	كونتكتور	Q0.1/K1M
٢	كونتكتور	Q0.2/K2M
٣	كونتكتور	Q0.3/K3M
٤	كونتكتور	Q0.4/K4M
٥	لمبة	Q1.1/K5M
٦	لمبة	Q2.1/K6M
٧	لمبة	Q1.2/K7M
٨	لمبة	Q2.2/K8M
٩	لمبة	Q1.3/K9M
١٠	لمبة	Q2.3/K10M
١١	لمبة	Q1.4/K11M
١٢	لمبة	Q2.4/K12M

البرنامج الرئيسى:

Network 1 Network Title

FIRIST MOTOR - RIGHT

I0.1 I0.0 Q0.2 Q0.1

Q0.1

Network 2

CALL FLASHER BY Q0.1

Q0.1 SBR_0 EN

I1.1 IN1

OUT1 Q1.1

OUT2 Q2.1

Network 3 Network Title

FIRIST MOTOR - LEFT

I0.2 I0.0 Q0.1 Q0.2

Q0.2

Network 4

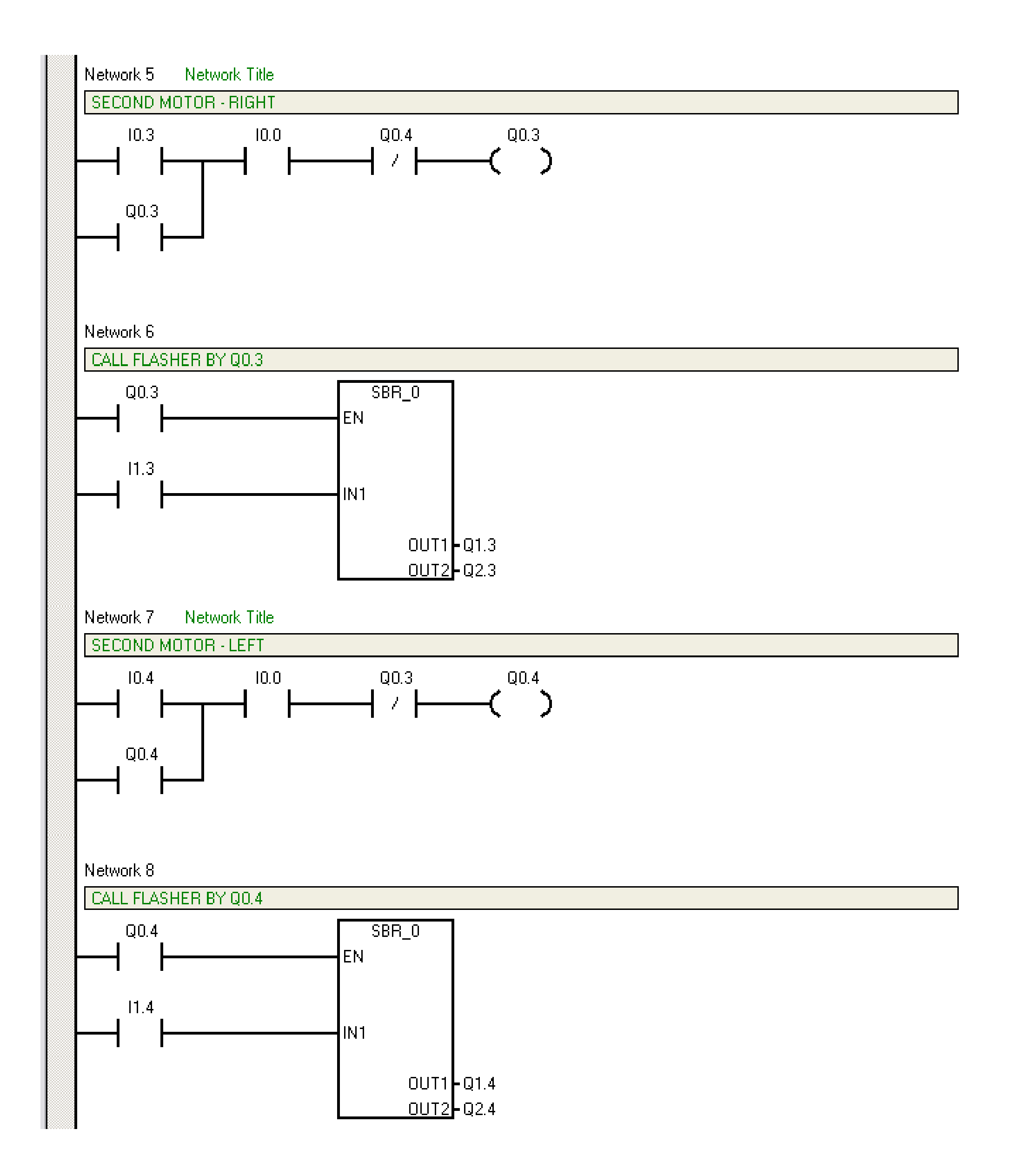
CALL FLASHER BY Q0.2

Q0.2 SBR_0 EN

I1.2 IN1

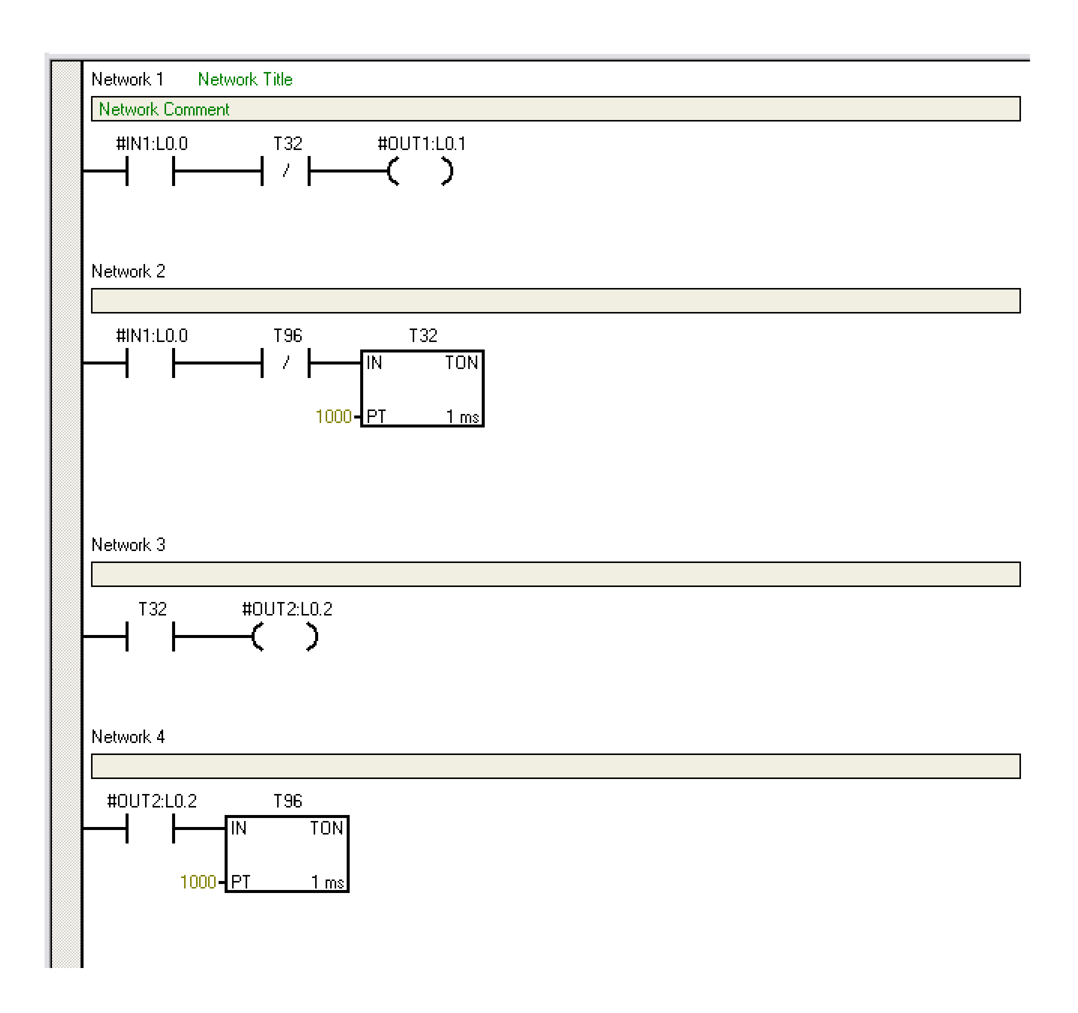
OUT1 Q1.2

OUT2 Q2.2



Network 5
Network Title
SECOND MOTOR - RIGHT
I0.3
I0.0
Q0.4
Q0.3
Q0.3
Network 6
CALL FLASHER BY Q0.3
Q0.3
SBR_0
EN
I1.3
IN1
OUT1
Q1.3
OUT2
Q2.3
Network 7
Network Title
SECOND MOTOR - LEFT
I0.4
I0.0
Q0.3
Q0.4
Q0.4
Network 8
CALL FLASHER BY Q0.4
Q0.4
SBR_0
EN
I1.4
IN1
OUT1
Q1.4
OUT2
Q2.4

البرنامج الفرعى:



- **شرح البرنامج الرئيسى:**

Network1: الجزء الخاص بحركة المحرك الأول لليمين

Network2: الجزء الخاص بالإتصال بالفلاشر الموجود فى البرنامج الفرعى الخاص بحركة المحرك الأول لليمين.

Network3: الجزء الخاص بحركة المحرك الأول لليسار

Network4: الجزء الخاص بالإتصال بالفلاشر الموجود فى البرنامج الفرعى الخاص بحركة المحرك الأول لليسار.

Network5: الجزء الخاص بحركة المحرك الثانى لليمين

Network6: الجزء الخاص بالإتصال بالفلاشر الموجود فى البرنامج الفرعى الخاص بحركة المحرك الثانى لليمين.

Network7: الجزء الخاص بحركة المحرك الثانى لليسار

Network8: الجزء الخاص بالإتصال بالفلاشر الموجود فى البرنامج الفرعى الخاص بحركة المحرك الثانى لليسار.

- **شرح البرنامج الفرعى:**

Network1: الجزء الخاص باللمبة الأولى.

Network2: الجزء الخاص بالمؤقت الزمنى T32.

Network3: الجزء الخاص باللمبة الثانية.

Network4: الجزء الخاص بالمؤقت الزمنى T96.

ملاحظة.

باستخدام الـ subroutine أصبح عدد الفروع ١٢بدلاً من ٢٠

الباب السابع

البوابات

- **شرح البوابـــــــــــــات.**
- **بوابـــــــــــــات "و".**
- **بوابـــــــــــــات "أو".**
- **بوابـــــــــــــات "نفى".**
- **أحجام البوابـــــــــــــات.**
- **المفاتيح المستخدمة مع البوابات.**
- **الأخـــــطاء الممكن التعرض لها.**
- **توضيحات بالرســــــــــم.**
- **تمـــــــــــــارين عملية.**

البوابات المنطقية:

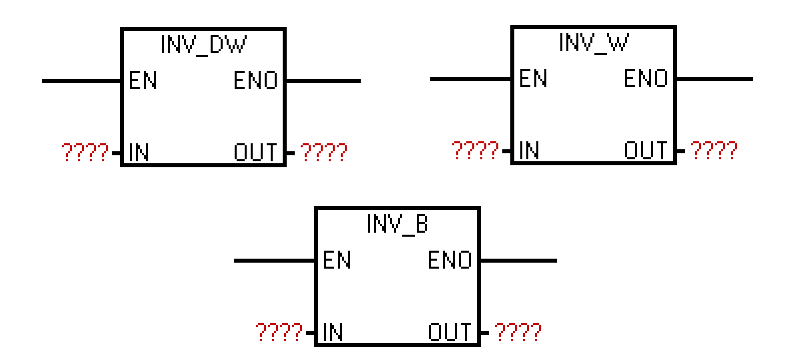
تستخدم البوابات فى البرامج العادية أو حتى التى تحتوى على عمليات حسابية.

أنواع البوابات:

INV_B, INV_W, INV_DW, WAND_B, WAND_W, WAND_DW, WOR_B, WOR_W, WOR_DW, WXOR_B, WXOR_W, WXOR_DW.

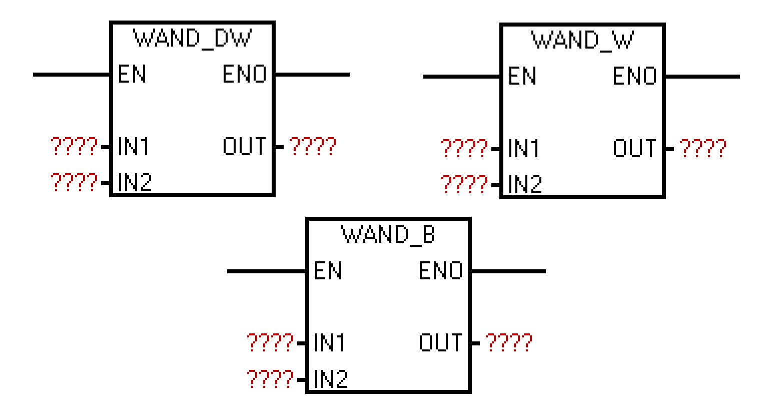
بوابات النفى "INVERT":

تقوم بتحويل الصفر إلى واحد و الواحد إلى صفر و تتواجد هذه العملية بحجم Byte, Word و Dword.



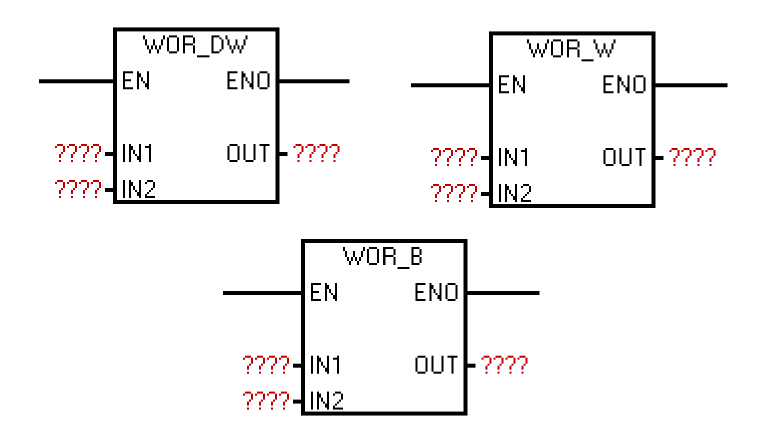
بوابات "AND":

تقوم بتطبيق البوابة المنطقية "و" بين مجموعتين و تتواجد هذه العملية بحجم Byte, Word و Dword.



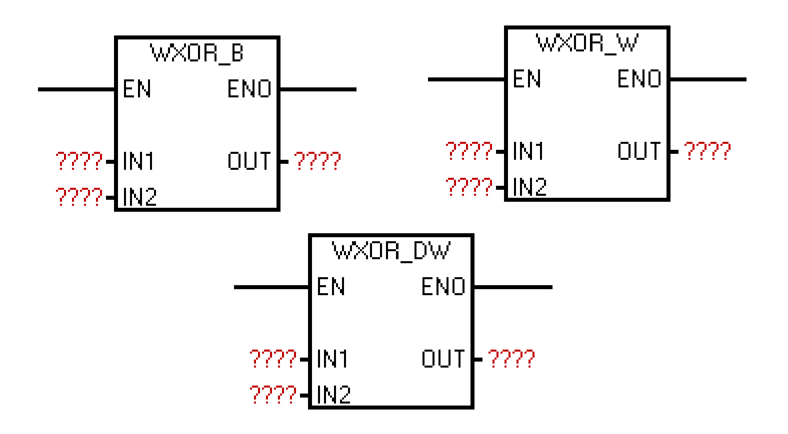
بوابات "OR":

تقوم بتطبيق البوابة المنطقية "أو" بين مجموعتين و تتواجد هذه العملية بحجم Byte, Word و Dword



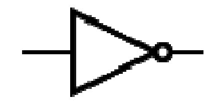
بوابات "XOR":

تقوم بتطبيق البوابة المنطقية "XOR" بين مجموعتين و تتواجد هذه العملية بحجم Byte, Word و Dword.



شرح البوابات:

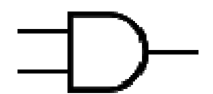
بوابات INV:



Input	Output
A	INV(A)
1	0
0	1

البوابة "نفى" تقوم بتحويل الواحد إلى صفر و الصفر إلى واحد.

بوابات AND:



Input		Output
A	B	A x B
0	0	0
0	1	0
1	0	0
1	1	1

البوابة "و": تصبح القيمة واحد عندما تتحقق كل القيم.

بوابات OR:



Input		Output
A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	1

البوابة "أو": تصبح القيمة واحد عندما تتحقق قيمة واحدة أو كل القيم.

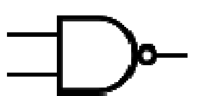
بوابات XOR:



Input		Output
A	B	A xor B
0	0	0
0	1	1
1	0	1
1	1	0

البوابة "أكسور": تصبح القيمة واحد عندما لا تتشابه كل القيم.

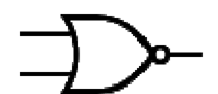
بوابات NAND:



Input		Output
A	B	A nand B
0	0	1
0	1	1
1	0	1
1	1	0

البوابة "ناند": تصبح القيمة واحد فى جميع الأحوال ماعدا فى حالة الوحايد.

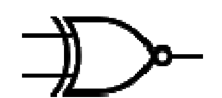
بوابات NOR:



Input		Output
A	B	A nor B
0	0	1
0	1	0
1	0	0
1	1	0

البوابة "نور": تصبح القيمة صفر فى جميع الأحوال ماعدا فى حالة الأصفار.

بوابات XNOR:



Input		Output
A	B	A xnor B
0	0	1
0	1	0
1	0	0
1	1	1

البوابة "أكسنور": تصبح القيمة واحد فى حالة تشابه القيم فقط أى فى حالة الوحايد والأصفار.

لا توجد بوابات مباشرة بأسم NAND, أو NOR, أو XOR لأنه يمكن التوصل إليها باستخدام البوابات AND أو OR أو XOR مع البوابة INV.

الشرح:

م	الأسم	الشرح	الشكل
١	INV_B	يقوم INV_B بعكس محتويات الـ Bit من الـ Byte IN إلى الـ Byte OUT.	INV_B EN ENO ????-IN OUT-????
٢	INV_W	يقوم INV_W بعكس محتويات الـ Bit من الـ Word IN إلى الـ Word OUT.	INV_W EN ENO ????-IN OUT-????
٣	INV_DW	يقوم INV_DW بعكس محتويات الـ Bit من الـ Dword IN إلى الـ Dword OUT.	INV_DW EN ENO ????-IN OUT-????
٤	WAND_B	يقوم AND_B بتنفيذ عملية "و" بين محتويات الـ Bits الخاصة بالـ Byte	WAND_B EN ENO ????-IN1 OUT-???? ????-IN2

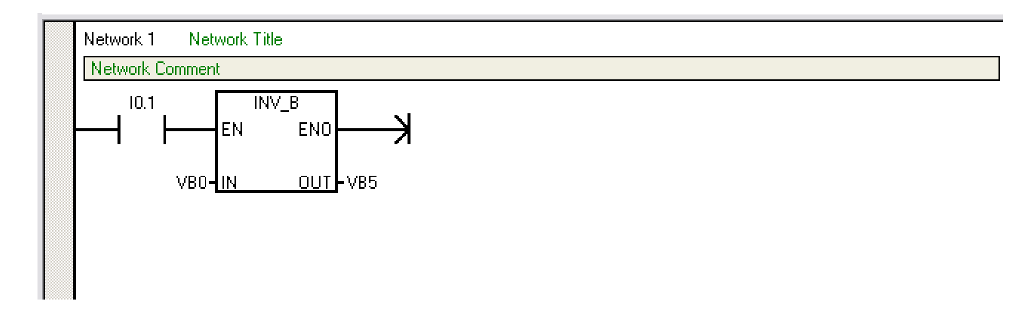
		IN1 و محتويات ال Bits الخاصة بال IN2 Byte.	
٥	WAND_W	يقوم AND_W بتنفيذ عملية "و" بين محتويات ال Bits الخاصة بال Word IN1 و محتويات ال Bits الخاصة بال IN2 Word.	WAND_W EN ENO ???? IN1 OUT ???? ???? IN2
٦	WAND_DW	يقوم AND_DW بتنفيذ عملية "و" بين محتويات ال Bits الخاصة بال IN1 Dword و محتويات ال Bits الخاصة بال IN2 Dword.	WAND_DW EN ENO ???? IN1 OUT ???? ???? IN2
٧	WOR_B	يقوم WOR_B بتنفيذ عملية "أو" بين محتويات ال Bits الخاصة بال Byte IN1 و محتويات ال Bits الخاصة بال IN2 Byte.	WOR_B EN ENO ???? IN1 OUT ???? ???? IN2

٨	WOR_W	يقوم WOR_W بتنفيذ عملية "أو" بين محتويات الـ Bits الخاصة بالـ Word IN1 و محتويات الـ Bits الخاصة بالـ Word IN2.	WOR_W EN ENO ????-IN1 OUT-???? ????-IN2
٩	WOR_DW	يقوم WOR_DW بتنفيذ عملية "أو" بين محتويات الـ Bits الخاصة بالـ Dword IN1 و محتويات الـ Bits الخاصة بالـ Dword IN2.	WOR_DW EN ENO ????-IN1 OUT-???? ????-IN2
١٠	WXOR_B	يقوم WXOR_B بتنفيذ عملية "أكسور" بين محتويات الـ Bits الخاصة بالـ Byte IN1 و محتويات الـ Bits الخاصة بالـ Byte IN2.	WXOR_B EN ENO ????-IN1 OUT-???? ????-IN2
١١	WXOR_W	يقوم WXOR_W بتنفيذ عملية "أكسور" بين محتويات الـ Bits الخاصة بالـ Word IN1 و	WXOR_W EN ENO ????-IN1 OUT-???? ????-IN2

		محتويات الـ Bits الخاصة بالـ Word IN2.	
١٢	WXOR_DW	يقوم WXOR_DW بتنفيذ عملية "أكسور" بين محتويات الـ Bits الخاصة بالـ Dword IN1 و محتويات الـ Bits الخاصة بالـ Dword IN2.	WXOR_DW EN ENO ????-IN1 OUT-???? ????-IN2

مثال:

تمرين باستخدام INV_B.



VB0

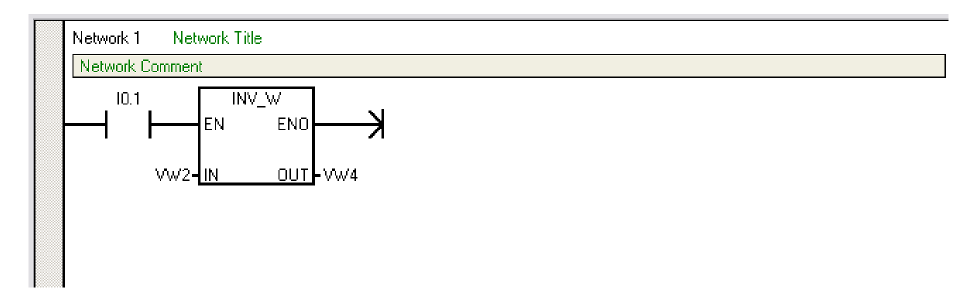
1	0	1	0	0	0	1	1

VB5

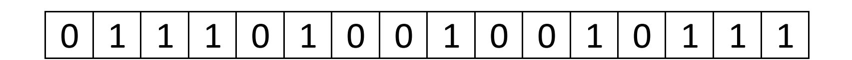
0	1	0	1	1	1	0	0

مثال:

تمرين باستخدام INV_W.



VW2



0	1	1	1	0	1	0	0	1	0	0	1	0	1	1	1

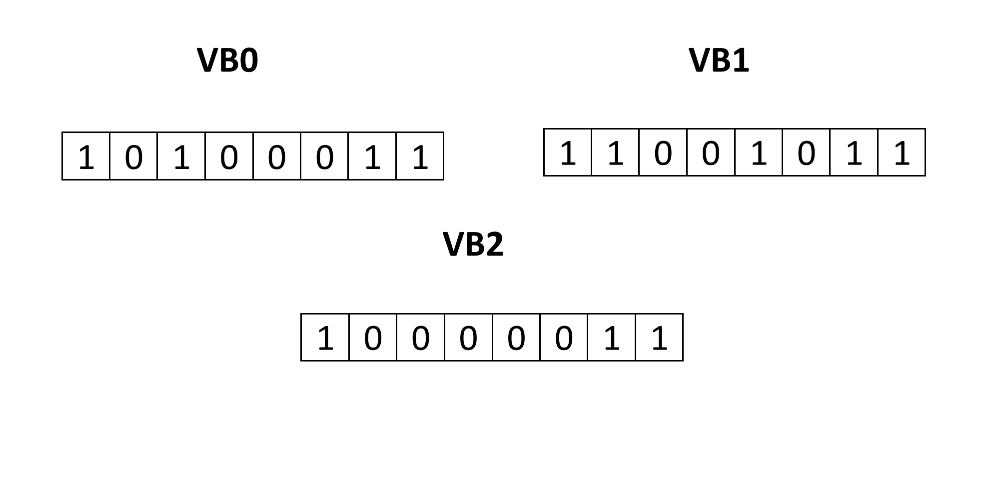
VW4

1	0	0	0	1	0	1	1	0	1	1	0	1	0	0	0

مثال:

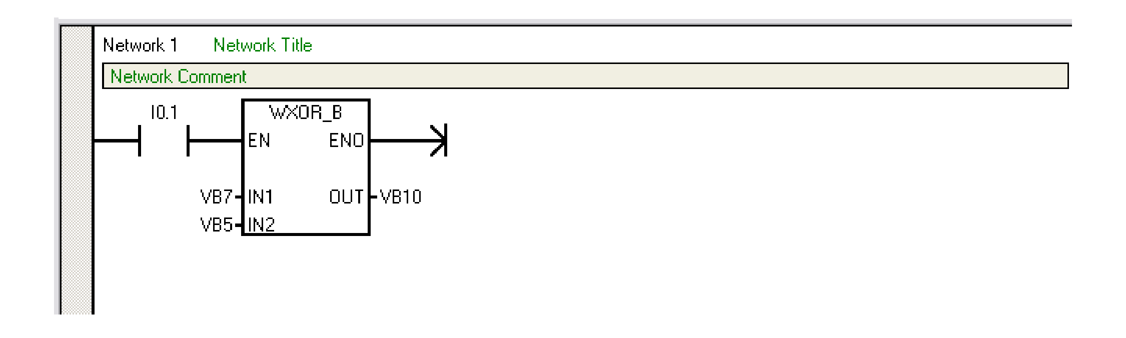
تمرين باستخدام WAND_B.

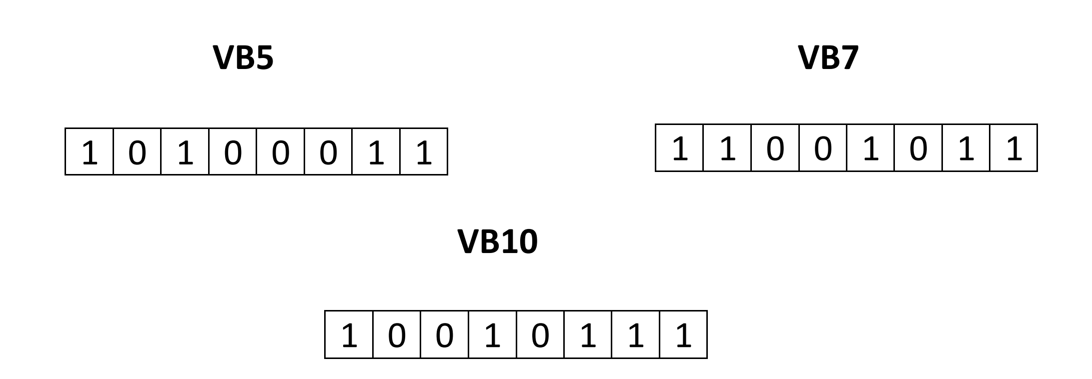
Network 1 Network Title
Network Comment
I0.1
WAND_B
EN ENO
VB0 IN1 OUT VB2
VB1 IN2



مثال:

تمرين باستخدام WXOR_B.

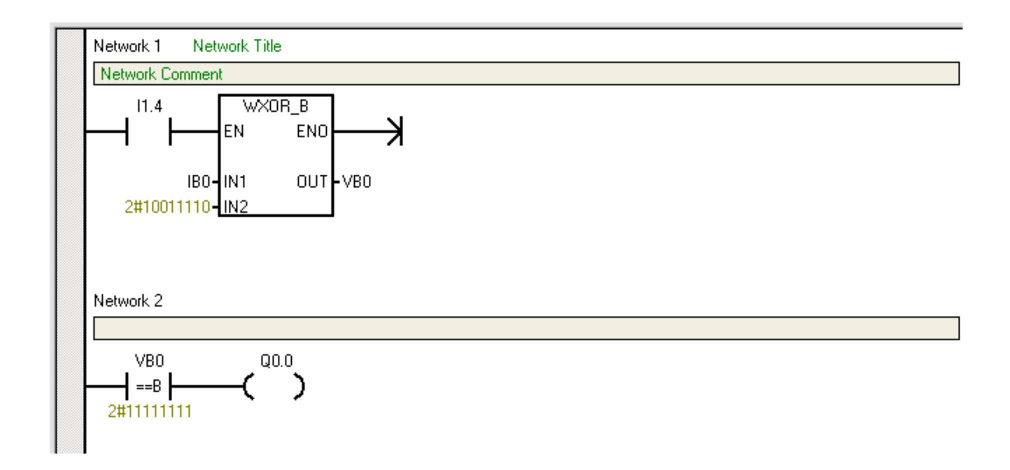




مثال عملى:

قم بتنفيذ دائرة تحكم منطقية لمحرك يعمل بشرط أن تكون حالة المفاتيح كالأتى:

I0.7	I0.6	I0.5	I0.4	I0.3	I0.2	I0.1	I0.0
1	0	0	1	1	1	1	0



:Network1

يتم تطبيق العملية "XOR" بين IB0 و 2#1001-1110 و وضع النتيجة فى VB0

:Network2

تعمل Q0.0 أذا كانت قيمة VB0 هى 2#1111-1111.

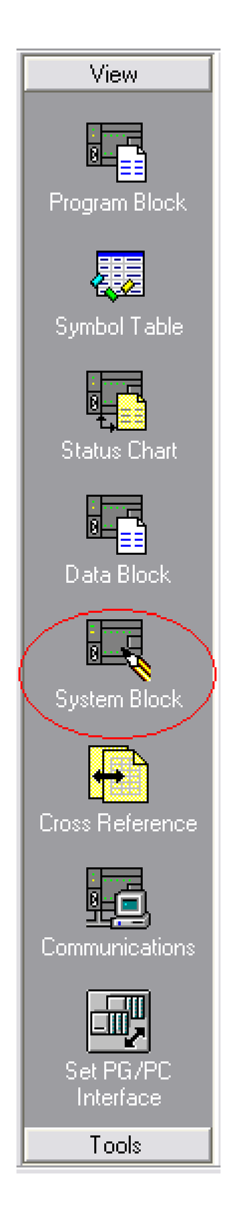
الباب الثامن

النظم العملية

- شرح النظـــــــــــــــــــــــــــــم العملية.
- صفحـــة الـ Communication Ports.
- صفحـــــــــــــــة Retentive Ranges.
- صفحـــــــــــــــــــــــــــــة Password.
- صفحة Output Tables DIGITAL.
- صفحــــة Input Filters DIGITAL.
- صفحــــــــــــة الـ Pulse Chatch Bits.
- صفحــــــــــة الـ Background Time.
- صفحـــــــــــــــة الـ Configure LED.
- صفحــــــــــــة الـ Increase Memory.
- المفاتيح المستخدمة فى صفحة النظـم العمليـة.
- الأخـــــــــــــطاء الممكن التعرض لها.

صفحة النظم العملية:

تستخدم صفحة "النظم العملية" الـ System Block لتحديد بعض المتغيرات الخاصة بالبرنامج والتى يمكن لها أن تغير فى طبيعة عمل البرنامج من حيث طريقة التشغيل.



طريقة استخدام صفحة "النظم العملية":

- يتم عمل البرنامج أولاً ولكن قبل تحميل البرنامج يتم تحديد المتغيرات بواسطة صفحة " النظم العملية " وبعد ذالك يتم تحميل البرنامج.

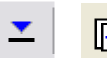
ملاحظة:

فى حاله رسم البرنامج و تحميله ثم تعديل بعض المتغيرات باستخدام صفحة " النظم العملية " دون تحميل البرنامج مرة أخرى فأن كل المتغيرات تعتبر غير فعالة ولذالك سيلحظ دائماً ظهور رسالة فى جميع الصفح الفرعية داخل صفحة "النظم العملية" وهى:

" Configuration parameters must be downloaded before they take effect"

أى أنه "لن يحدث أى تغير قبل تنفيذ عملية التحميل"

المفاتيح المستخدمة فى صفحة الرموز.



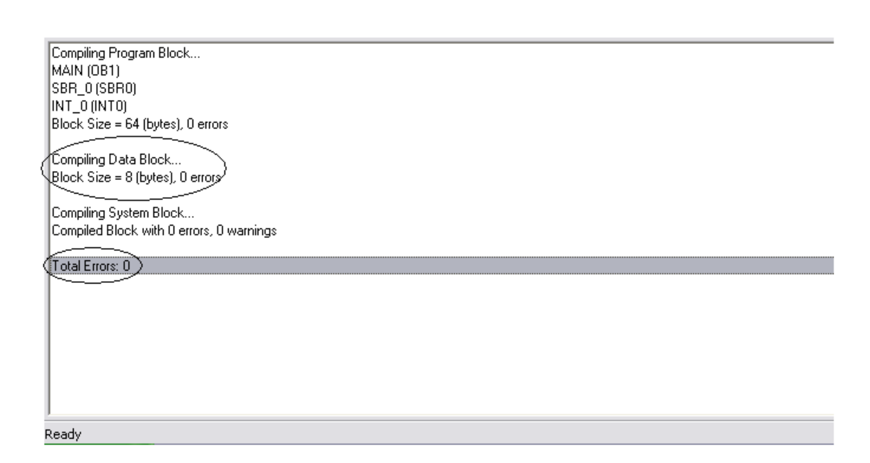
:Download

بالضغط على هذا المفتاح يتم تحميل البرنامج كما سبق و شرحنا و تصبح جميع المتغيرات المحددة بواسطة صفحة "النظم العملية" متاحة.

Compile all

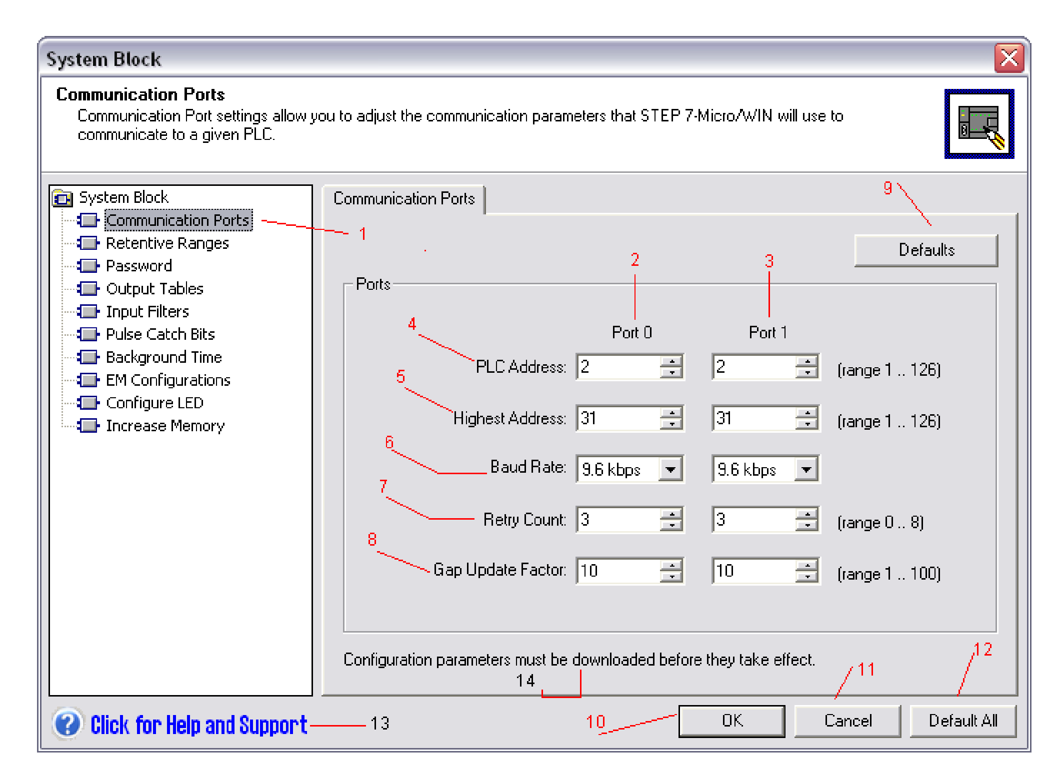
بالضغط على هذا المفتاح تظهر عدد الأعطال الخاصة بالبرنامج لكى يتم تجنبها قبل تطبيق أمر التحميل.

بالضغط على compile all تظهر هذه الرسالة التى توضح الأخطاء إذا وجدت:



١- صفحة Communication Ports.

هى صفحة لتحديد المتغيرات الخاصة بالتواصل بين وحدة ال PLC و جهاز التحكم أى أن كان نوعه.



١- صفحة ال Communication Ports.

٢- المقصود بكلمة Port0 هى البيانات الخاصة بوحدة ال PLC.

٣- المقصود بكلمة Port1 هى البيانات الخاصة بالوحدة الإضافية.

٤- المقصود بكلمة PLC Address هو العنوان الخاص بوحدة ال PLC وغالباً ما يكون أثنان.

٥- المقصود بكلمة Highest Address هو أقصى أسم لوحدات ال PLC يمكن للحاسب الألى التواصل معهم.

٦- المقصود بكلمة Baud Rate هى سرعة تبادل البيانات بين الحاسب الألى و وحدة ال PLC.

٧- المقصود بكلمة Retry Count هى عدد محاولات أعادة التواصل بين الحاسب الألى و وحدة ال PLC.

٨- المقصود بكلمة Gap Update Factor هى الفراغات بين وحدات ال PLC ويفضل بأن يكون عدد ليس بكبير.

٩- المقصود بكلمة Default هى أعادة المتغيرات الخاصة بصفحة Communication Ports فقط إلى طبيعتها.

١٠- المقصود بكلمة Ok هو تفعيل جميع المتغيرات التى نفذت و لكن بعد تحميل البرنامج.

١١- المقصود بكلمة Cancel هو إلغاء التعديلات و غلق الصفحة.

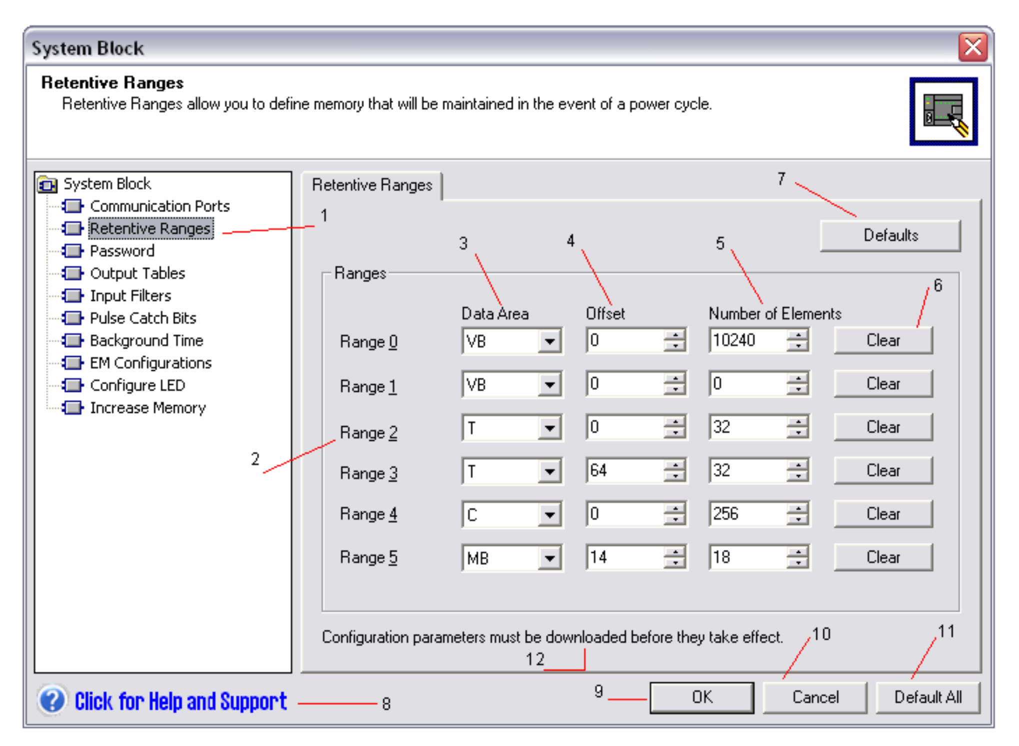
١٢- المقصود بكلمة Default All هى أعادة المتغيرات الخاصة بجميع صفح System Block إلى طبيعتها.

١٣- المقصود بكلمة Help هى صفحة للتوضيح و المساعدة خاصة بصفحة Communication Ports فقط.

١٤- لتفعيل أى من المتغيرات التى سوف يتم تطبيقها بواسطة صفحة Communication Ports يجب تحميل البرنامج.

٢- صفحة Retentive Ranges.

تستخدم هذه الصفحة لتطبيق أمر Retentive على ريليهات أو مؤقتات زمنية أو عدادات لكى يحتفظوا بقيمتهم.



١- صفحة ال Retentive Ranges.

٢- المقصود بكلمة Range هى المجموعات التى سوف يتم التعامل معها.

٣- المقصود بكلمة Data Area هى القائمة التى تحتوى على العناوين المستخدمة فى صفحة ال Retentive Ranges.

٤- المقصود بكلمة Offset هو أسم العنوان الذى سوف يتم تطبيق مبدأ ال Retentive عليه.

٥- المقصود بكلمة Number of Elements هى عدد العناوين التى سوف يتم تطبيق مبدأ ال Retentive عليه بدايتاً من العنوان المدون فى ال Offset.

٦- المقصود بكلمة Clear هو محو جميع التعديلات الخاصة بالصف المجاور لها.

٧- المقصود بكلمة Default هى أعادة المتغيرات الخاصة بصفحة Retentive Ranges فقط إلى طبيعتها.

٨- المقصود بكلمة Help هى صفحة للتوضيح و المساعدة خاصة بصفحة Retentive Ranges فقط.

٩- المقصود بكلمة Ok هو تفعيل جميع المتغيرات التى نفذت و لكن بعد تحميل البرنامج.

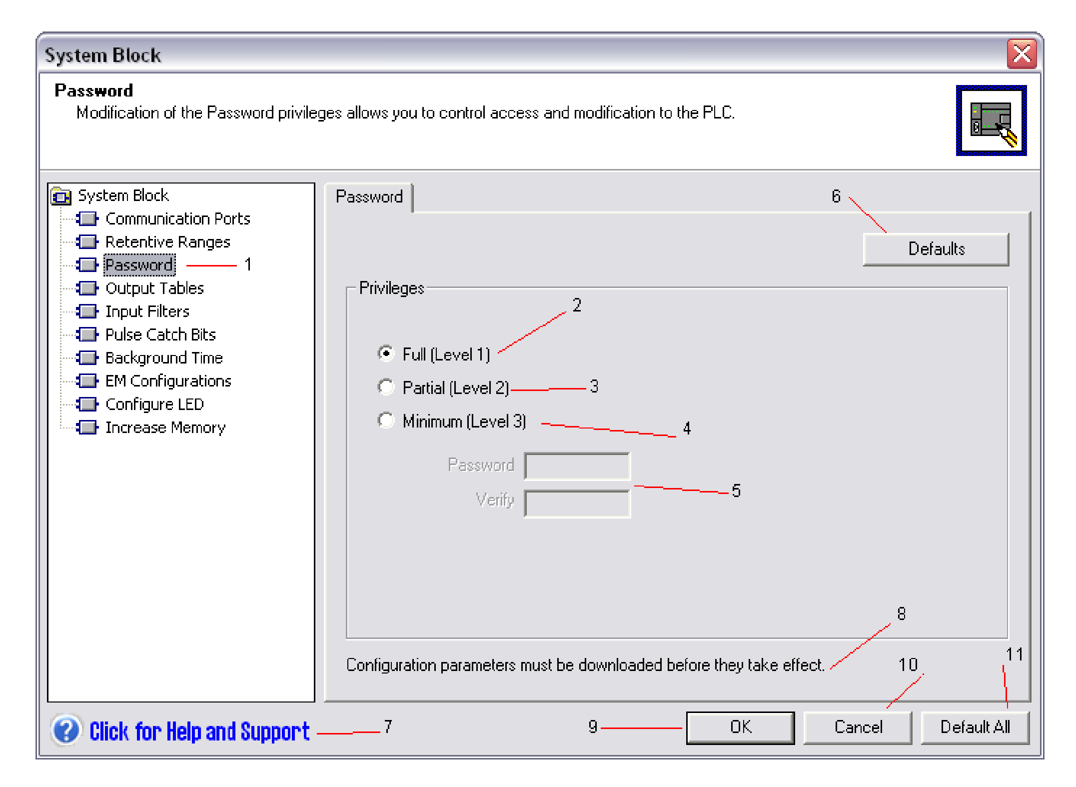
١٠- المقصود بكلمة Cancel هو إلغاء التعديلات و غلق الصفحة.

١١- المقصود بكلمة Default All هى أعادة المتغيرات الخاصة بجميع صفح Retentive Ranges إلى طبيعتها.

١٢- لتفعيل أى من المتغيرات التى سوف يتم تطبيقها بواسطة صفحة Retentive Ranges يجب تحميل البرنامج.

٣- صفحة Password.

تستخدم هذه الصفحة لوضع كلمة مرور لتحديد بعض الأعمال.



١- صفحة ال Password.

٢- المقصود بكلمة Full level هو أقل مستوى من مستويات كلمات المرور حيث أنه لا تأثير لكلمة المرور الخاصة بهذا المستوى.

٣- المقصود بكلمة Partial level هو ثاني مستوى من مستويات كلمات المرور حيث أنه لا يمكن تحميل البرنامج أو تطبيق أمر Force أثناء تفعيل كلمة المرور الخاصة بهذا المستوى.

٤- المقصود بكلمة Minimum level هو أقوى مستوى من مستويات كلمات المرور حيث أنه لا يمكن تحميل البرنامج download أو تطبيق أمر Force أو أمر Upload أثناء تفعيل كلمة المرور الخاصة بهذا المستوى.

٥- المقصود بهذا الفراغ هو المكان الذى يتم كتابة كلمة المرور بداخله مرتين.

٦- المقصود بكلمة Default هى أعادة المتغيرات الخاصة بصفحة Password فقط إلى طبيعتها.

٧- المقصود بكلمة Help هي صفحة للتوضيح و المساعدة خاصة بصفحة Password فقط.

٨- لتفعيل أى من المتغيرات التي سوف يتم تطبيقها بواسطة صفحة Password يجب تحميل البرنامج.

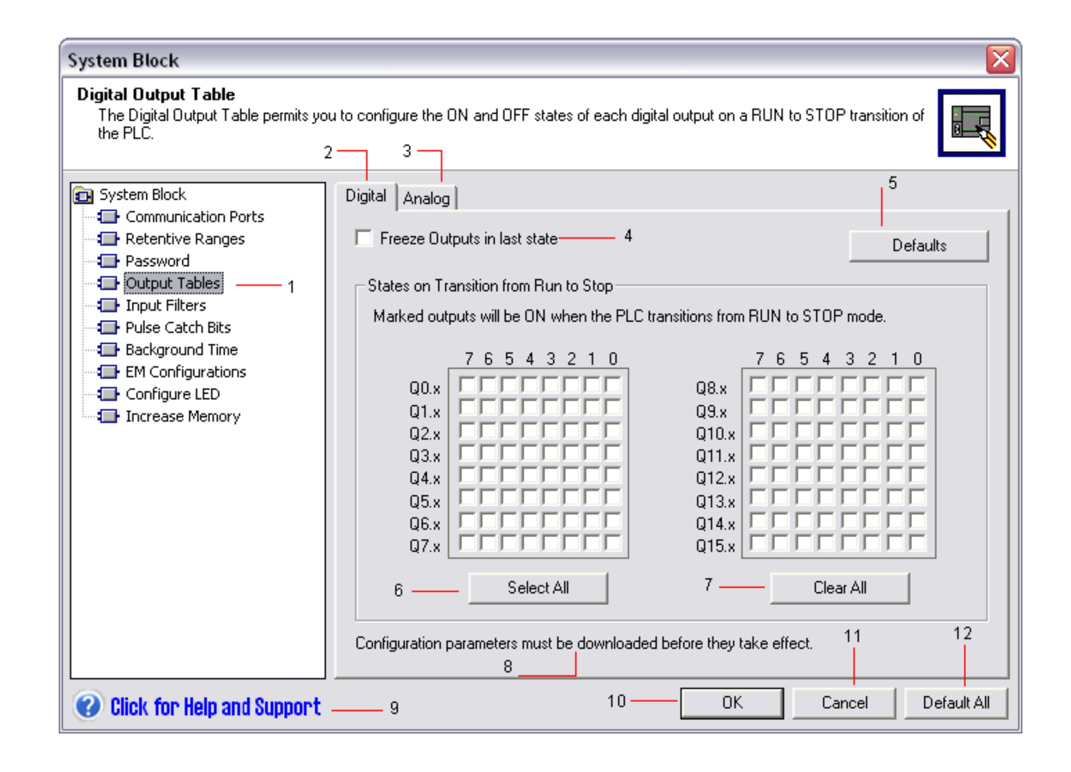
٩- المقصود بكلمة Ok هو تفعيل جميع المتغيرات التي نفذت و لكن بعد تحميل البرنامج.

١٠- المقصود بكلمة Cancel هو إلغاء التعديلات و غلق الصفحة.

١١- المقصود بكلمة Default All هي أعادة المتغيرات الخاصة بجميع صفح System block إلى طبيعتها.

٤- صفحة Output Tables DIGITAL.

تستخدم هذه الصفحة لإختيار المخرجات Digital المراد أن تحتفظ بأخر حالة لها عند تحويل وضع وحدة الـ PLC من عمل إلى إيقاف.



١- صفحة الـ Output Tables DIGITAL.

٢- المقصود بكلمة Digital هى صفحة خاصة بالخرج الرقمى.

٣- المقصود بكلمة Analog هى صفحة خاصة بالخرج التناظرى.

٤- المقصود بكلمة Freeze Output in last state هو تثبيت الخرج على أخر حاله له.

٥- المقصود بكلمة Default هى أعادة المتغيرات الخاصة بصفحة Output Tables DIGITAL فقط إلى طبيعتها.

٦- المقصود بكلمة Select All هو إختيار جميع المخرجات لكى يطبق عليها أمر التثبيت Freeze.

٧- المقصود بكلمة Clear All هو عدم إختيار جميع المخرجات لكى لا يطبق عليها أمر التثبيت Freeze.

٨- لتفعيل أى من المتغيرات التى سوف يتم تطبيقها بواسطة صفحة Output Tables DIGITAL يجب تحميل البرنامج.

٩- المقصود بكلمة Help هى صفحة للتوضيح و المساعدة خاصة بصفحة Output Tables DIGITAL فقط.

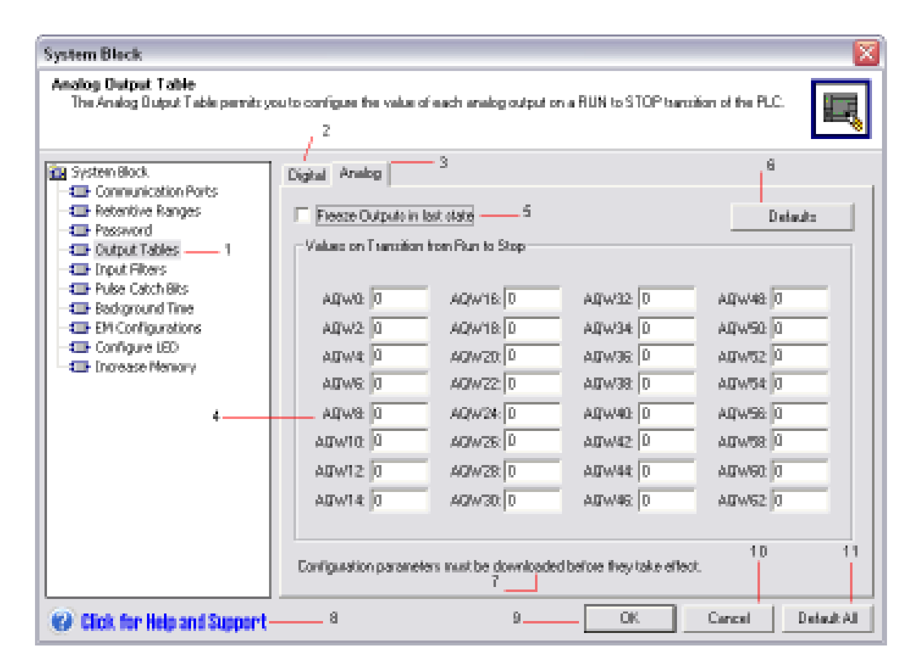
١٠- المقصود بكلمة Ok هو تفعيل جميع المتغيرات التى نفذت و لكن بعد تحميل البرنامج.

١١- المقصود بكلمة Cancel هو إلغاء التعديلات و غلق الصفحة.

١٢- المقصود بكلمة Default All هى أعادة المتغيرات الخاصة بجميع صفحات System block إلى طبيعتها.

٥- صفحة Output Tables ANALOG.

تستخدم هذه الصفحة لأختيار المخرجات التناظرية المراد أن تحتفظ بأخر حالة "قيمة" لها عند تحويل وضع وحدة ال PLC من عمل إلى إيقاف.



١- صفحة ال Output Tables ANALOG.

٢- المقصود بكلمة Digital هى صفحة خاصة بالخرج الرقمى.

٣- المقصود بكلمة Analog هى صفحة خاصة بالخرج التناظرى.

٤- المقصود بكلمة AQW هو أسم الخرج التناظرى.

٥- المقصود بكلمة Freeze Output in last state هو تثبيت الخرج على أخر قيمة له.

٦- المقصود بكلمة Default هى أعادة المتغيرات الخاصة بصفحة Output Tables Analog فقط إلى طبيعتها.

٧- لتفعيل أى من المتغيرات التى سوف يتم تطبيقها بواسطة صفحة Output Tables ANALOG يجب تحميل البرنامج.

٨- المقصود بكلمة Help هى صفحة للتوضيح و المساعدة خاصة بصفحة Output Tables ANALOG فقط.

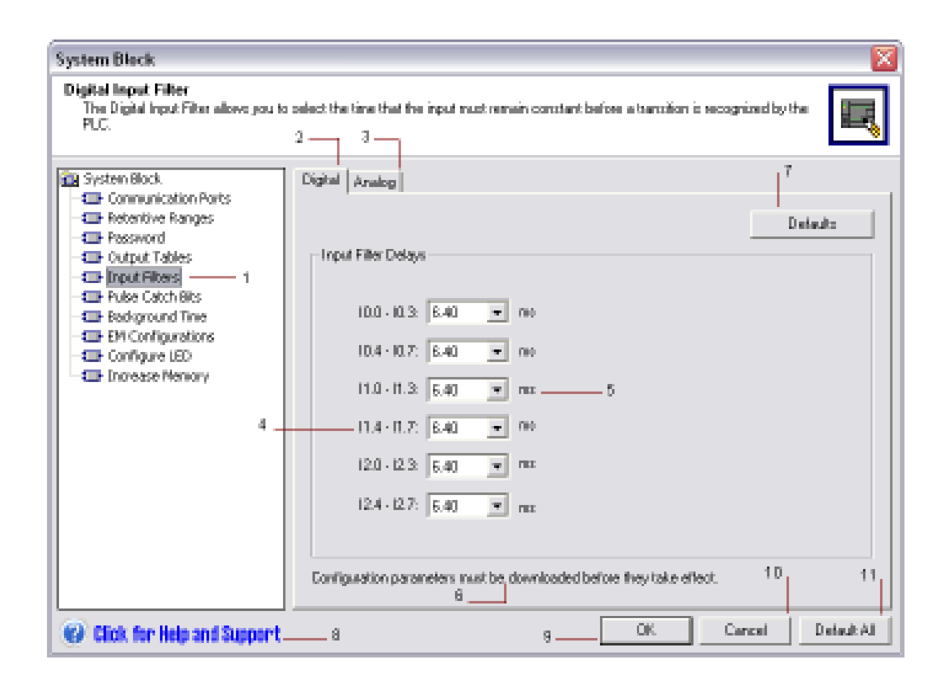
٩- المقصود بكلمة Ok هو تفعيل جميع المتغيرات التى نفذت و لكن بعد تحميل البرنامج.

١٠- المقصود بكلمة Cancel هو إلغاء التعديلات و غلق الصفحة.

١١- المقصود بكلمة Default All هى أعادة المتغيرات الخاصة بجميع صفح System block إلى طبيعتها.

٦- صفحة Input Filters DIGITAL.

تستخدم هذه الصفحة لاختيار زمن الإشارة الخاصة بالمداخلات Digital.



١- صفحة الـ Input Filters DIGITAL.

٢- المقصود بكلمة Digital هى صفحة خاصة بالخرج الرقمى.

٣- المقصود بكلمة Analog هى صفحة خاصة بالخرج التناظرى.

٤- المقصود به مجموعة من المدخلات التى سوف يطبق عليها أمر Input Filter.

٥- المقصود به الزمن المراد تطبيقه.

٦- لتفعيل أى من المتغيرات التى سوف يتم تطبيقها بواسطة صفحة Input Filters DIGITAL يجب تحميل البرنامج.

٧- المقصود بكلمة Default هى أعادة المتغيرات الخاصة بصفحة Input Filters DIGITAL فقط إلى طبيعتها.

٨- المقصود بكلمة Help هى صفحة للتوضيح و المساعدة خاصة بصفحة Input Filters DIGITAL فقط.

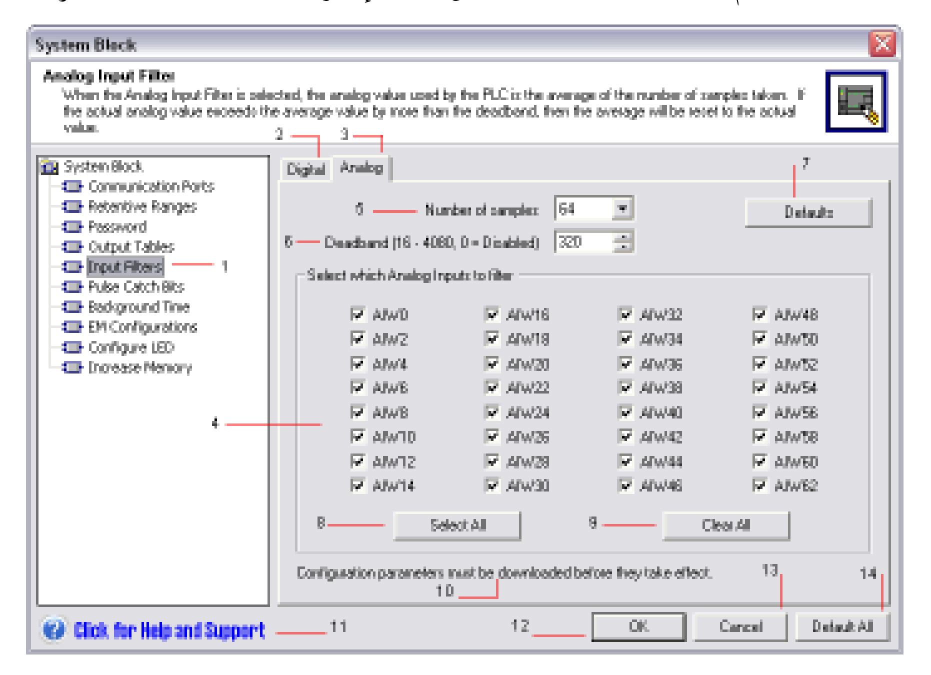
٩- المقصود بكلمة Ok هو تفعيل جميع المتغيرات التى نفذت و لكن بعد تحميل البرنامج.

١٠- المقصود بكلمة Cancel هو إلغاء التعديلات و غلق الصفحة.

١١- المقصود بكلمة Default All هى أعادة المتغيرات الخاصة بجميع صفح System block إلى طبيعتها.

٧- صفحة Input Filters ANALOG.

تستخدم هذه الصفحة للتحديد المسبق لقيمة الإشارة الخاصة بالمدخلات التناظرية.



١- صفحة ال Input Filters ANALOG.

٢- المقصود بكلمة Digital هى صفحة خاصة بالخرج الرقمى.

٣- المقصود بكلمة Analog هى صفحة خاصة بالخرج .

٤- المقصود به مجموعة من المدخلات التى سوف يطبق عليها أمر Input Filter.

٥- المقصود بكلمة Number of samples حيث يمكن تحديد قيمة مسبقة للمدخلات التناظرية.

٦- المقصود بكلمة Deadband حيث يتم تحديد التغيير الذى يسمح بقبوله من المدخلات التناظرية.

٧- المقصود بكلمة Default هى أعادة المتغيرات الخاصة بصفحة Input Filters ANALOG فقط إلى طبيعتها.

٨- المقصود بكلمة Select All هو اختيار جميع المدخلات لكى يتم تفعيلها.

٩- المقصود بكلمة Clear All هو عدم اختيار أى من المدخلات لكى لا يطبق عليها أمر Filter.

١٠- لتفعيل أى من المتغيرات التى سوف يتم تطبيقها بواسطة صفحة Input Filters ANALOG يجب تحميل البرنامج.

١١- المقصود بكلمة Help هى صفحة للتوضيح و المساعدة خاصة بصفحة Input Filters ANALOG فقط.

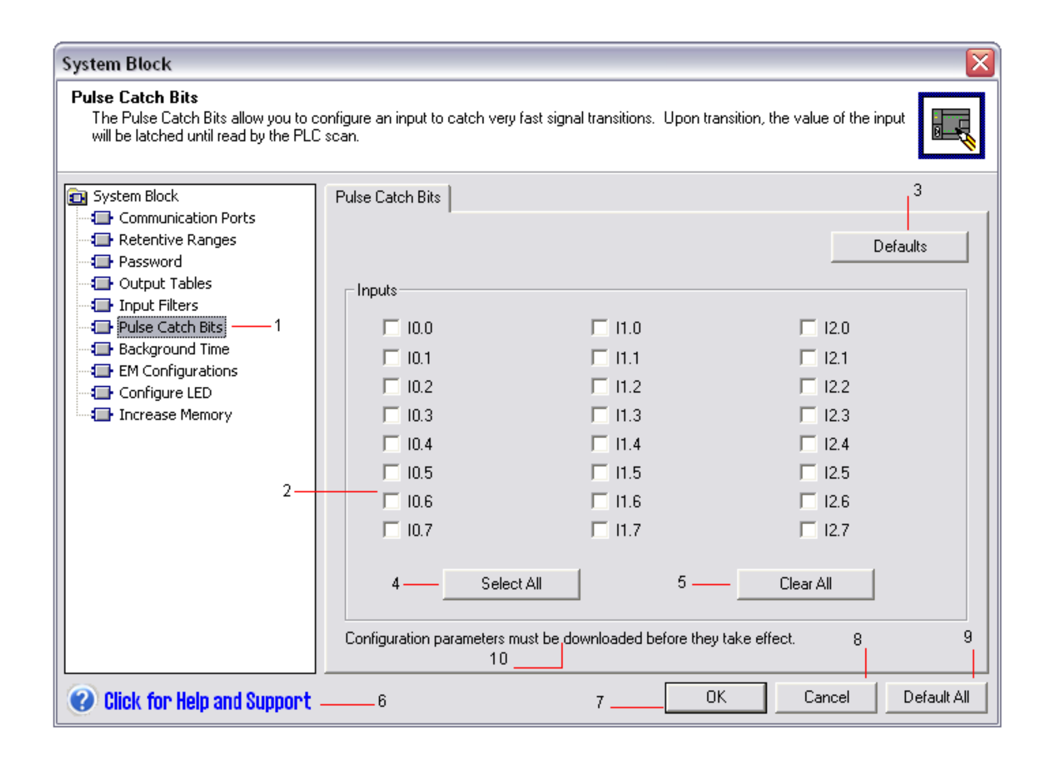
١٢- المقصود بكلمة Ok هو تفعيل جميع المتغيرات التى نفذت و لكن بعد تحميل البرنامج.

١٣- المقصود بكلمة Cancel هو إلغاء التعديلات و غلق الصفحة.

١٤- المقصود بكلمة Default All هى أعادة المتغيرات الخاصة بجميع صفح Input Filters ANALOG إلى طبيعتها.

٨- صفحة Pulse catch Bits.

تستخدم هذه الصفحة للاحتفاظ بزمن الإشارة الخاصة بالمدخلات حتى يستكمل دورة كاملة إضافية.



١- صفحة الـ Pulse Catch Bits.

٢- حيث يتم اختيار المدخلات المراد التعامل معها.

٣- المقصود بكلمة Default هى أعادة المتغيرات الخاصة بصفحة Pulse Catch Bits فقط إلى طبيعتها.

٤- المقصود بكلمة Select All هو اختيار جميع المدخلات لكى يتم تفعيلها.

٥- المقصود بكلمة Clear All هو عدم اختيار أى من المدخلات لكى لا يطبق عليها أمر Catch.

٦- المقصود بكلمة Help هى صفحة للتوضيح و المساعدة خاصة بصفحة Pulse Catch Bits فقط.

٧- المقصود بكلمة Ok هو تفعيل جميع المتغيرات التى نفذت و لكن بعد تحميل البرنامج.

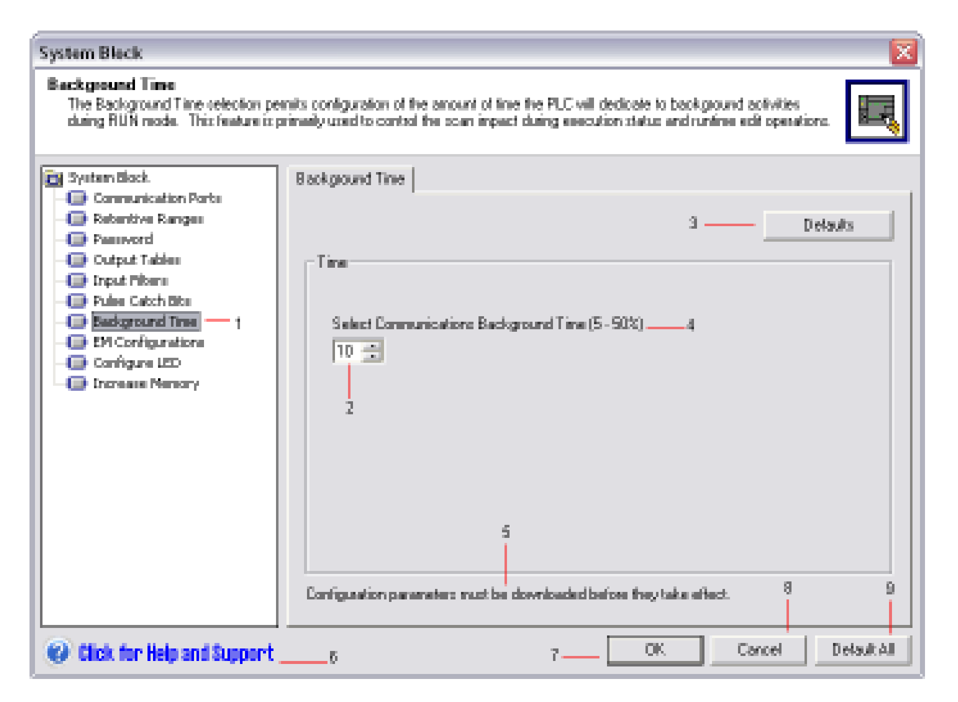
٨- المقصود بكلمة Cancel هو إلغاء التعديلات و غلق الصفحة.

٩- المقصود بكلمة Default All هى أعادة المتغيرات الخاصة بجميع صفح System block إلى طبيعتها.

١٠- لتفعيل أى من المتغيرات التى سوف يتم تطبيقها بواسطة صفحة Pulse Catch Bits يجب تحميل البرنامج.

٩- صفحة **Background Time.**

تستخدم هذه الصفحة للتحكم فى سرعة الأمر التوضيحى program status.



١ – صفحة الـ Background Time.

٢ – المقصود به هو النسبة المئوية للتحكم بالزمن الفعلى.

٣ – المقصود بكلمة Default هى أعادة المتغيرات الخاصة بصفحة Background Time فقط إلى طبيعتها.

٤ – يمكن التحكم بالزمن حتى أنه يمكن أبطاء الزمن الحقيقى من ٥% إلى ٥٠%.

٥ – لتفعيل أى من المتغيرات التى سوف يتم تطبيقها بواسطة صفحة Background Time يجب تحميل البرنامج.

٦ – المقصود بكلمة Help هى صفحة للتوضيح و المساعدة خاصة بصفحة Background Time فقط.

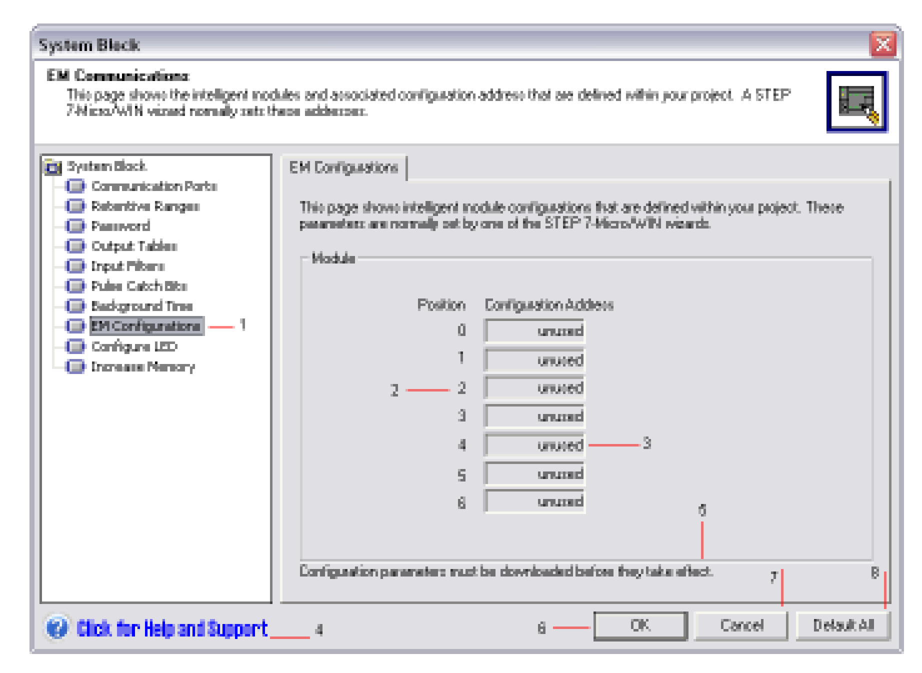
٧ – المقصود بكلمة Ok هو تفعيل جميع المتغيرات التى نفذت و لكن بعد تحميل البرنامج.

٨- المقصود بكلمة Cancel هو إلغاء التعديلات و غلق الصفحة.

٩- المقصود بكلمة Default All هى أعادة المتغيرات الخاصة بجميع صفح System block إلى طبيعتها.

١٠- صفحة EM Configurations.

تستخدم هذه الصفحة للتحكم فى سرعة الأمر التوضيحى program status.



١- صفحة الـ EM Configurations.

٢- المقصود به هو تسلسل وحدة المدخلات أو المخرجات الإضافية.

٣- المقصود به هو توضيح أذا كان وحدة المدخلات أو المخرجات الإضافية مستخدمة أم غير مستخدمة.

٤- لتفعيل أى من المتغيرات التى سوف يتم تطبيقها بواسطة صفحة EM Configurations يجب تحميل البرنامج.

٥- المقصود بكلمة Help هى صفحة للتوضيح و المساعدة خاصة بصفحة EM Configurations فقط.

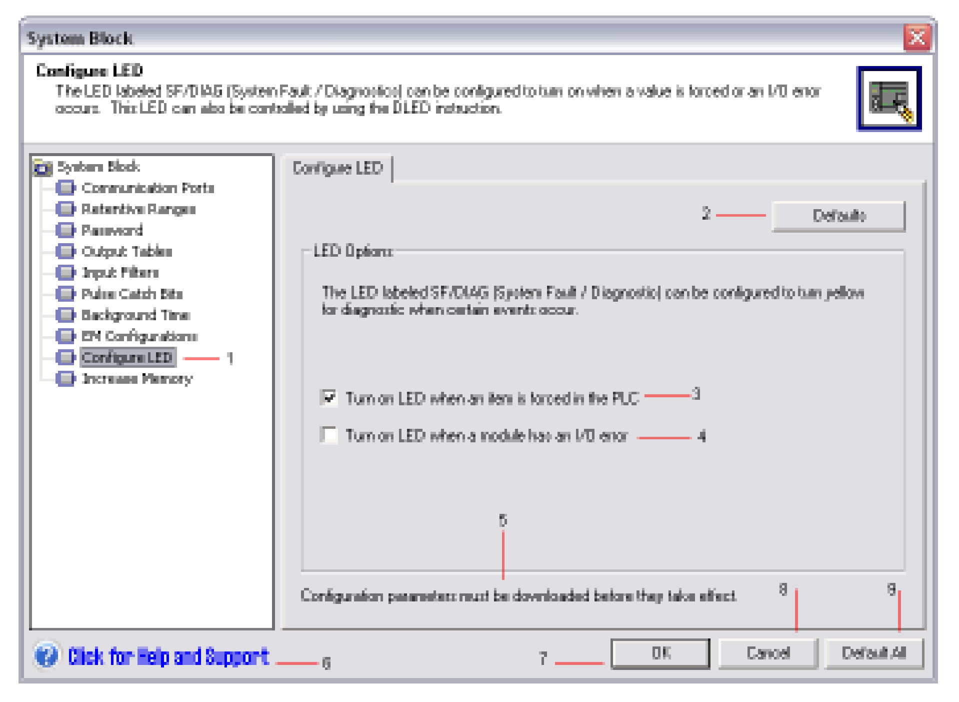
٦- المقصود بكلمة Ok هو تفعيل جميع المتغيرات التى نفذت و لكن بعد تحميل البرنامج.

٧- المقصود بكلمة Cancel هو إلغاء التعديلات و غلق الصفحة.

٨- المقصود بكلمة Default All هى أعادة المتغيرات الخاصة بجميع صفح System block إلى طبيعتها.

١١- صفحة Configure LED.

تستخدم هذه الصفحة للتحكم فى سرعة الأمر التوضيحى program status.



١- صفحة ال Configure LED.

٢- المقصود بكلمة Default هى أعادة المتغيرات الخاصة بصفحة Configure LED فقط إلى طبيعتها.

٣- المقصود به هو أنه سوف تضىء لمبة الأعطال فى حالة استخدام أمر Force.

٤- المقصود به هو أنه سوف تضىء لمبة الأعطال فى حالة وجود أعطال خاصة بالمداخلات أو بالمخرجات.

٥- لتفعيل أى من المتغيرات التى سوف يتم تطبيقها بواسطة صفحة Configure LED يجب تحميل البرنامج.

٦- المقصود بكلمة Help هى صفحة للتوضيح و المساعدة خاصة بصفحة Configure LED فقط.

٧- المقصود بكلمة Ok هو تفعيل جميع المتغيرات التى نفذت و لكن بعد تحميل البرنامج.

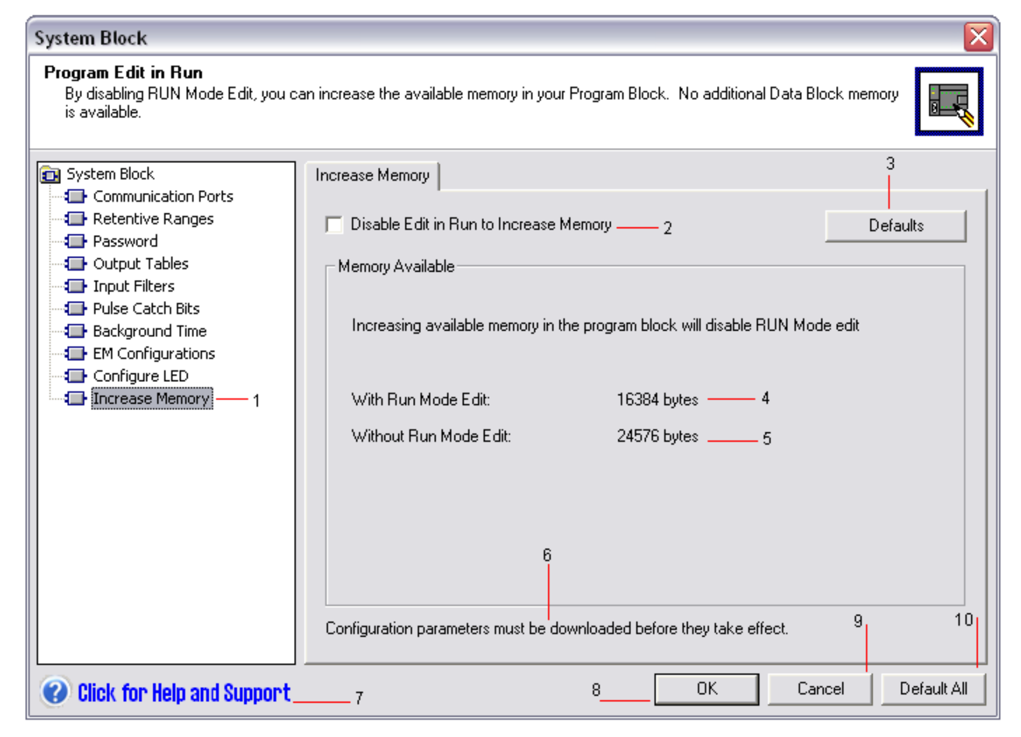
٨- المقصود بكلمة Cancel هو إلغاء التعديلات و غلق الصفحة.

٩- المقصود بكلمة Default All هى أعادة المتغيرات الخاصة بجميع صفح System block إلى طبيعتها.

١٢- صفحة Increase Memory.

تستخدم هذه الصفحة للتحكم فى حجم الذاكرة سواء بالتعديل فى البرنامج أثناء تشغيل أو أثناء إيقاف الـ PLC:

- حيث أنة فى حالة تعديل البرنامج أثناء العمل فهذا يعنى أنه يجب على الذاكرة احتواء البرنامج القديم حتى يتم تفعيل البرنامج الجديد.
- حيث أنة فى حالة تعديل البرنامج أثناء التوقف فهذا يعنى أنه يجب على الذاكرة محو البرنامج القديم ثم تحميل البرنامج الجديد.



١– صفحة الـ Increase Memory.

٢– المقصود به هو عدم التعديل فى البرنامج أثناء عمل وحدة الـ PLC.

٣– المقصود بكلمة Default هى أعادة المتغيرات الخاصة بصفحة Increase Memory فقط إلى طبيعتها.

٤– المقصود به هو أن التعديل فى البرنامج أثناء عمل وحدة الـ PLC يوفر فى الذاكرة المستخدمة فى البرمجة.

٥– المقصود به هو أن التعديل فى البرنامج أثناء وقوف وحدة الـ PLC لا يوفر فى الذاكرة المستخدمة فى البرمجة.

٦– لتفعيل أى من المتغيرات التى سوف يتم تطبيقها بواسطة صفحة Increase Memory يجب تحميل البرنامج.

٧– المقصود بكلمة Help هى صفحة للتوضيح و المساعدة خاصة بصفحة Increase Memory فقط.

٨- المقصود بكلمة Ok هو تفعيل جميع المتغيرات التى نفذت و لكن بعد تحميل البرنامج.

٩- المقصود بكلمة Cancel هو إلغاء التعديلات و غلق الصفحة.

١٠- المقصود بكلمة Default All هى أعادة المتغيرات الخاصة بجميع صفح System block إلى طبيعتها.

الباب التاسع

الريليهات الخاصة

- شرح الريليهات الخاصة.
- أنواع الريليهات الخاصة.
- صفحة الريليهات الخاصة.
- تعليقات الريليهات الخاصة.
- رموز الريليهات الخاصة.
- المفاتيح المستخدمة مع الريليهات الخاصة.
- تمارين عملية للتوضيح.

الريليهات الخاصة:

تعتبر الريليهات الخاصة هى عبار عن ذاكرة خاصة داخل وحدة الـ PLC تعمل بطرق مختلفة مقارنة بطرق عمل الريليهات الداخلية العادية.

مقارنة بين النوعين

م	الفرق	الريليهات الداخلية العادية	الريليهات الخاصة
١	كيفية العمل	تعمل حسب شروط وضعت فى البرنامج	تعمل حسب شروط وضعت فى الوحدة وغير قابلة للتعديل
٢	كيفية الإيقاف	تقف حسب شروط وضعت فى البرنامج	تقف حسب شروط وضعت فى الوحدة وغير قابلة للتعديل

تعمل الريليهات الخاصة أى الريليهات المبرمجة بطريقة محددة مسبقاً من قبل وحدة البرمجة, فمثلاً توجد:

- ريليهات تعمل مرة كل نصف ثانية.
- ريليهات تعمل لمرة واحدة فقط.
- ريليهات تعمل مرة كل دورة للبرنامج.
- ريليهات تعمل فى حالة أعطال.
- ريليهات تعمل مرة كل نصف ساعة.
- ريليهات تعمل فى حالة فصل البطارية.
- ريليهات تعمل فى حالة أجبار أى عنوان على العمل أو الإيقاف.

شرح لبعض مفاتيح الريليهات الخاصة:

م	أسم الريليه	شرح طريقة عمل الريليه الخاص
١	SM0.0	يغلق هذا المفتاح دائماً.
٢	SM0.1	يغلق هذا المفتاح فقط فى أول Cycle.
٣	SM0.2	يغلق هذا المفتاح فى حالة فقدان أى قيمة من قيم الـ Retentive.
٤	SM0.3	يغلق هذا المفتاح لزمن Cycle عندما تتحول وحدة الـ PLC إلى العمل.
٥	SM0.4	يغلق هذا المفتاح لثلاثين ثانية ويفتح لثلاثين ثانية أخرى.
٦	SM0.5	يغلق هذا المفتاح لنصف ثانية ويفتح لنصف ثانية أخرى.
٧	SM0.6	يغلق هذا المفتاح لزمن cycle ويفتح لزمن cycle أخر.
٨	SM0.7	يغلق هذا المفتاح عندما يكون المفتاح فى وضع run و يفتح عندما يكون فى وضع term.
٩	SM1.0	يغلق هذا المفتاح عندما يكون الناتج الخاص بأى عملية حسابية يساوى صفر.
١٠	SM1.1	يغلق هذا المفتاح عندما يكون الناتج أكبر من حجم الذاكرة المستخدمة.
١١	SM1.2	يغلق هذا المفتاح عندما يكون الناتج الخاص بأى عملية حسابية سالب.
١٢	SM1.3	يغلق هذا المفتاح عندما يتم القسمة على صفر فى أى عملية حسابية.
١٣	SM4.7	يغلق هذا المفتاح عندما يتم تطبيق أمر Force على أى عنوان.
١٤	SM5.0	يغلق هذا المفتاح عندما تحدث أى أعطال بخصوص وحدة المدخلات أو المخرجات.
١٥	SM1.6	يغلق هذا المفتاح عندما يتم تحويل رقم BCD غير صحيح إلى Binary.

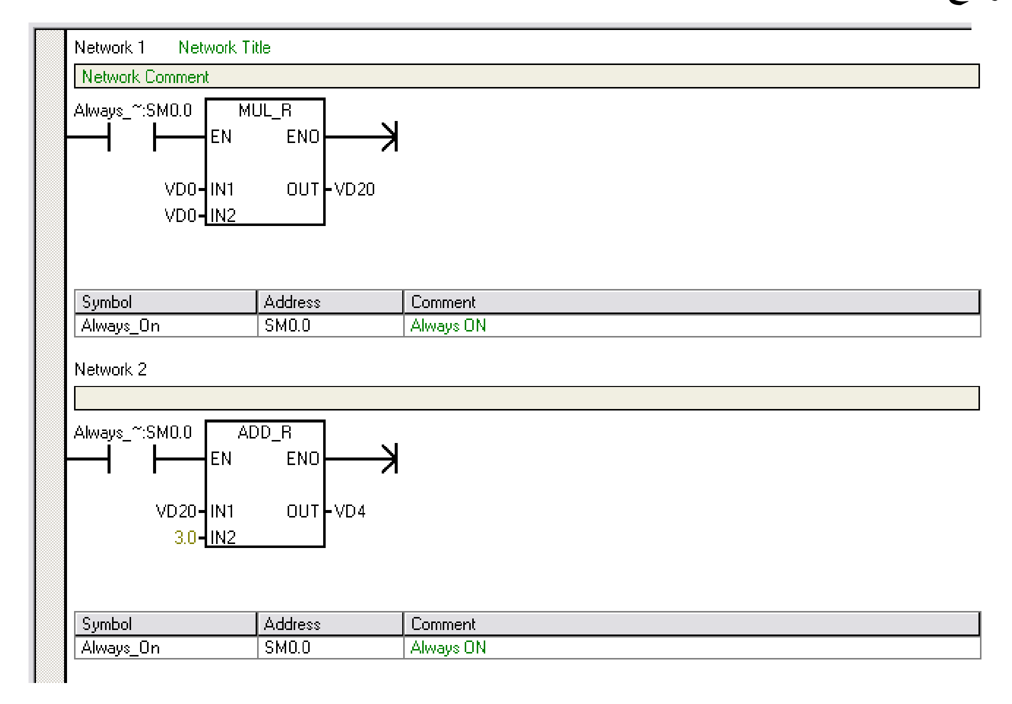
١٦	SM1.7	يغلق هذا المفتاح عندما لا يمكن تحويل رقم ASCII إلى ما يماثله فى الـ Hexadecimal.
١٧	SMB6	تحتوى على رقم الـ CPU.
١٨	SMB28	تحتوى على قيمة الـ Analog Adjustment الأول.
١٩	SMB29	تحتوى على قيمة الـ Analog Adjustment الثانى.
٢٠	SMW22	يحتوى على قيمة الـ Scan time الخاصة بأخر Cycle.
٢١	SMW24	يحتوى على أقل قيمة للـ Scan time تم الوصول لها أثناء البرنامج.
٢٢	SMW26	يحتوى على أكبر قيمة للـ Scan time تم الوصول لها أثناء البرنامج.

تمارين عملية:

قم تنفيذ برنامج لتمثيل المعادلة التالية: ص = $س^{٢}$ + ٣

عدد الدخل	**نوع الدخل**	**أسم الدخل**
لا يوجد	لا يوجد	لا يوجد
عدد العمليات الحسابية	**نوع العمليات الحسابية**	**أسم العمليات الحسابية**
١	ADD_R	ADD_R
٢	MUL_R	MUL_R
عدد المتغيرات	**نوع المتغيرات**	**أسم المتغيرات**
١	Dword, Real	VD0 (س)
٢	Dword, Real	VD4 (ص)
٣	Dword, Real	VD20
عدد الخرج	**نوع الخرج**	**أسم الخرج**
لا يوجد	لا يوجد	لا يوجد

البرنامج:



ملاحظة:

فبدلاً من وضع مفاتيح يمكن استخدام مفتاح من ضمن مفاتيح الريليهات الخاصة الذى يغلق دائماً.

تمارين عملية:

قم بتنفيذ برنامج لتمثيل المعادلة التالية بحيث أن تعمل الأصوات التحذيرية بطريقة متقطعة فى حالة:

- القسمة على صفر.
- ناتج سالـب.
- ناتج صـفر.
- ناتج أكبر من الذاكرة.

المعادلة:

$$ص = ٢س^{٢} + ٤س + ٣$$

المدخلات:

عدد الدخل	نوع الدخل	أسم الدخل
لا يوجد	لا يوجد	لا يوجد

العمليات الحسابية:

عدد العمليات الحسابية	نوع العمليات الحسابية	أسم العمليات الحسابية
١	ADD_R	ADD_R
٢	MUL_R	MUL_R

الريليهات الخاصة:

عدد الريليهات الخاصة	نوع الريليهات الخاصة	أسم الريليهات الخاصة
١	Bit	SM0.0
٢	Bit	SM0.5
٣	Bit	SM1.0
٤	Bit	SM1.1
٥	Bit	SM1.2
٦	Bit	SM1.3

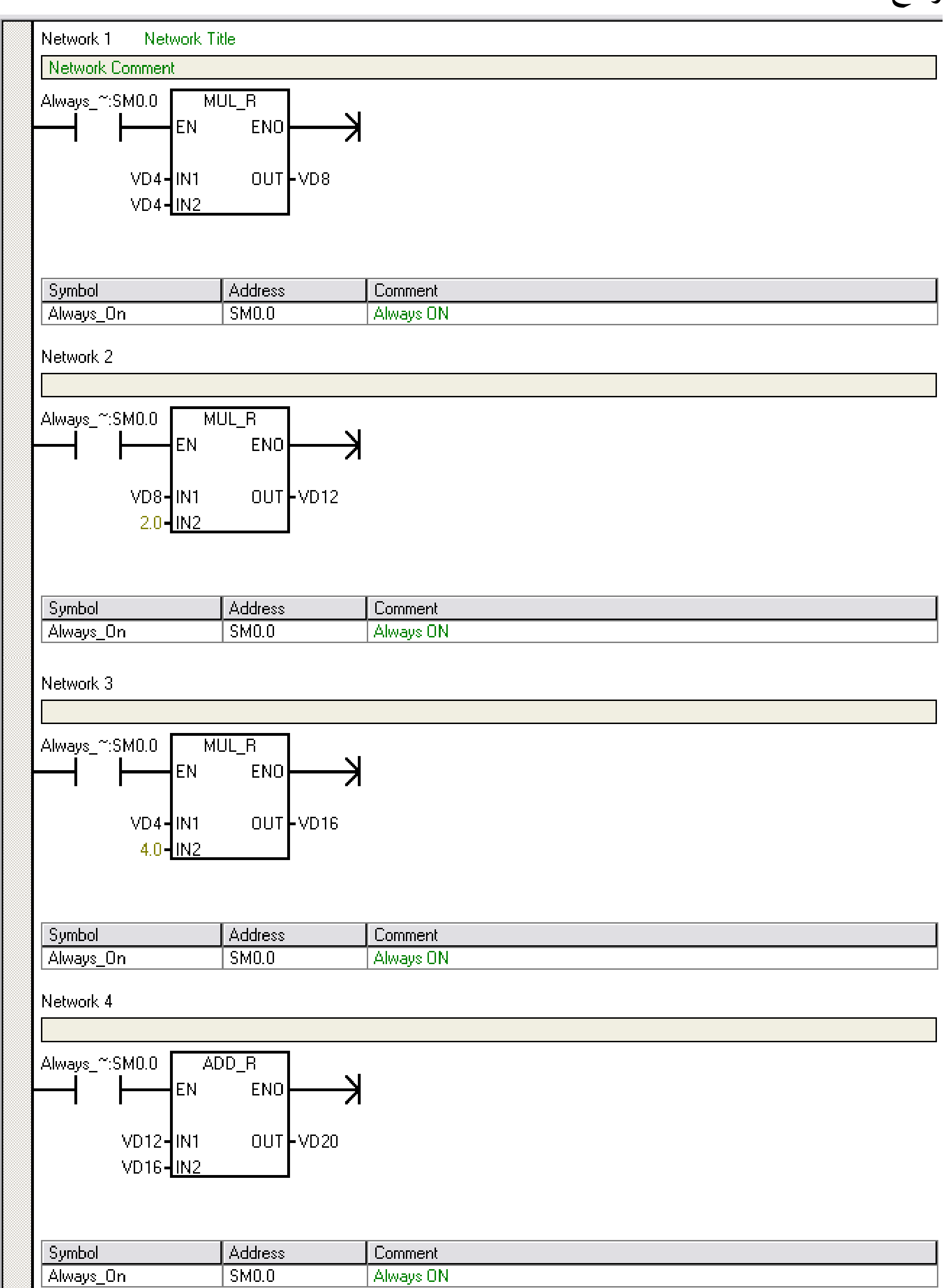
المتغيرات:

عدد المتغيرات	نوع المتغيرات	أسم المتغيرات
١	Dword, Real	VD0 (ص)
٢	Dword, Real	VD4 (س)
٣	Dword, Real	VD8 (س٢)
٤	Dword, Real	VD12 (٢س٢)
٥	Dword, Real	VD16 (٤س)
٦	Dword, Real	VD20 (٢س٢ + ٤س)

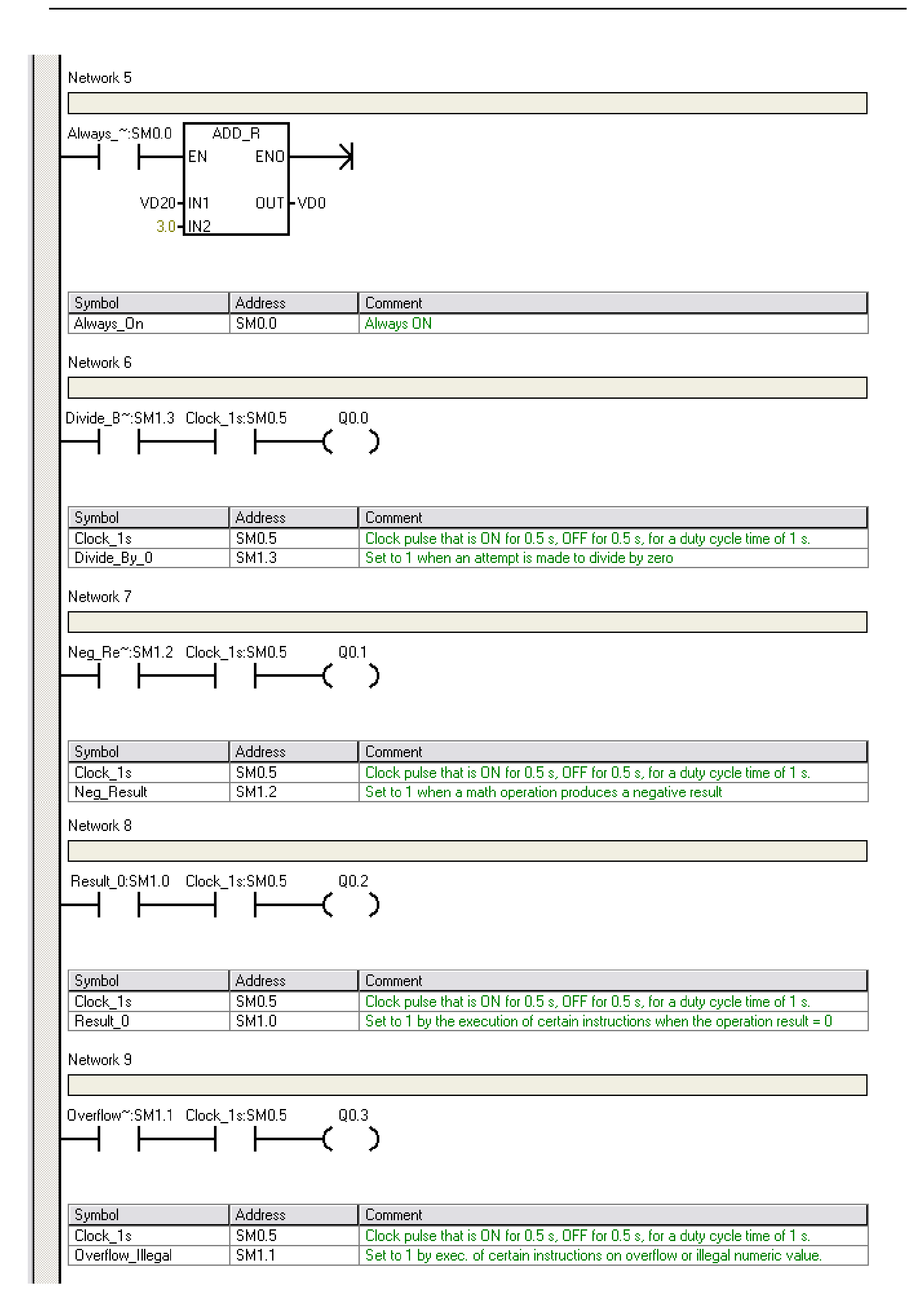
المخرجات:

عدد الخرج	نوع الخرج	أسم الخرج
١	إنذار (القسمة على صفر).	Q0.0/K1M
٢	إنذار (ناتج سالب).	Q0.1/K2M
٣	إنذار (ناتج صفر).	Q0.2/K3M
٤	إنذار (ناتج أكبر من الذاكرة).	Q0.3/K4M

البرنامج:



Network 1 Network Title
Network Comment
Always_~:SM0.0
MUL_R
EN ENO
VD4 IN1 OUT VD8
VD4 IN2
Symbol Address Comment
Always_On SM0.0 Always ON
Network 2
Always_~:SM0.0
MUL_R
EN ENO
VD8 IN1 OUT VD12
2.0 IN2
Symbol Address Comment
Always_On SM0.0 Always ON
Network 3
Always_~:SM0.0
MUL_R
EN ENO
VD4 IN1 OUT VD16
4.0 IN2
Symbol Address Comment
Always_On SM0.0 Always ON
Network 4
Always_~:SM0.0
ADD_R
EN ENO
VD12 IN1 OUT VD20
VD16 IN2
Symbol Address Comment
Always_On SM0.0 Always ON



Symbol	Address	Comment
Always_On	SM0.0	Always ON

Symbol	Address	Comment
Clock_1s	SM0.5	Clock pulse that is ON for 0.5 s, OFF for 0.5 s, for a duty cycle time of 1 s.
Divide_By_0	SM1.3	Set to 1 when an attempt is made to divide by zero

Symbol	Address	Comment
Clock_1s	SM0.5	Clock pulse that is ON for 0.5 s, OFF for 0.5 s, for a duty cycle time of 1 s.
Neg_Result	SM1.2	Set to 1 when a math operation produces a negative result

Symbol	Address	Comment
Clock_1s	SM0.5	Clock pulse that is ON for 0.5 s, OFF for 0.5 s, for a duty cycle time of 1 s.
Result_0	SM1.0	Set to 1 by the execution of certain instructions when the operation result = 0

Symbol	Address	Comment
Clock_1s	SM0.5	Clock pulse that is ON for 0.5 s, OFF for 0.5 s, for a duty cycle time of 1 s.
Overflow_Illegal	SM1.1	Set to 1 by exec. of certain instructions on overflow or illegal numeric value.

الشرح:

- **الجزء الخاص بالمعادلة الرياضية:**

:Network1

حيث يقوم بضرب قيمة VD4 فى VD4 ثم يضع الناتج فى VD8.

:Network2

حيث يقوم بضرب قيمة VD8 فى 2.0 ثم يضع الناتج فى VD12.

:Network3

حيث يقوم بضرب قيمة VD4 فى 4.0 ثم يضع الناتج فى VD16.

:Network4

حيث يقوم بجمع قيمة VD12 مع VD16 ثم يضع الناتج فى VD20.

:Network5

حيث يقوم بجمع قيمة VD12 مع 3.0 ثم يضع الناتج فى VD0.

- **الجزء الخاص بالريليهات الخاصة:**

:Network6

فى حاله القسمة على صفر سوف يعمل الإنذار Q0.0 بطريقة متقطعة.

:Network7

فى حاله أن الناتج قيمة سالبة سوف يعمل الإنذار Q0.1 بطريقة متقطعة.

:Network8

فى حاله أن الناتج يساوى صفر سوف يعمل الإنذار Q0.2 بطريقة متقطعة.

:Network9

فى حاله خطاء فى القيمة الرقمية للذاكرة سوف يعمل الإنذار Q0.3 بطريقة متقطعة.

الباب العاشر

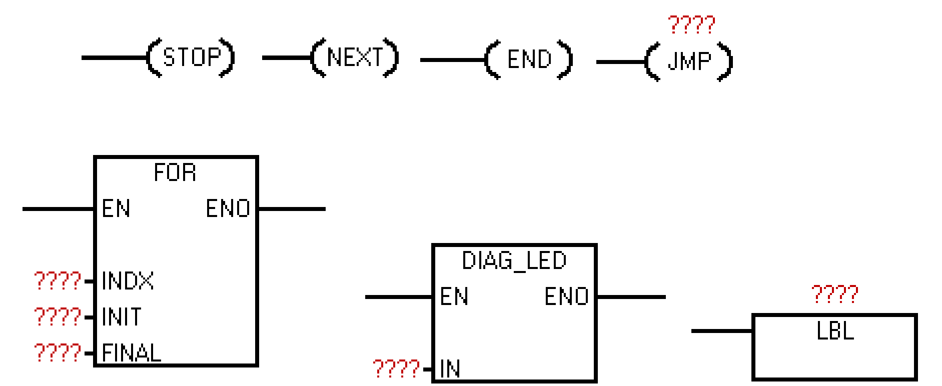
برامج التحكم

- شرح برامـــــــــج التحكم.
- محتويات برامـــــــج التحكم.
- شرح عمليـــــــــــة END.
- شرح عمليـــــــــة STOP.
- شرح عمليــــــــــة RET.
- شرح عمليـــة DIAG_LED.
- شرح عمليـــــة JMP/LBL.
- شرح عمليـــة FOR/NEXT.
- الأخـــــطاء الممكن التعرض لها.
- تمـــــــــارين عملية للتوضيح.

برامج التحكم:

تستخدم بعض الأوامر التالية للمساعدة فى البرمجة وليس بالشرط كجزء أساسى فى البرنامج.

END – STOP – JMP/LBL – FOR/NEXT – DIAGLED – RET



م	الأسم	الشرح	الشكل
١	END	يستخدم أمر END للتحكم بتنفيذ جزء معين حيث أنه يتحكم فى حجم الـ Cycle.	END
٢	STOP	يستخدم أمر STOP لإيقاف وحدة الـ PLC.	STOP
٣	DIAG_LED	يستخدم أمر DIAG_LED للتحكم بلمبة الـ S.F.	DIAG_LED EN ENO ???? IN

٤	JMP/LBL	يستخدم أمر JMP/LBL للتحكم بتنفيذ أو إلغاء جزء معين فى وسط الـ Cycle.	???? (JMP) ???? LBL
٥	FOR/NEXT	يستخدم أمر FOR/NEXT للتحكم بقراءة جزء معين فى البرنامج أكثر من مرة قبل الأنتهاء من الـ Cycle.	(NEXT) FOR EN ENO ????-INDX ????-INIT ????-FINAL
٦	RET	يستخدم أمر RET للخروج من صفحة البرمجة الفرعية قبل الأنتهاء من تنفيذ البرنامج الفرعى بالكامل.	(RET)

شرح كل نوع ورسم تمرين عملى للتوضيح

١- END:

كما سبق وذكرنا أن أمر END يستخدم للتحكم بتنفيذ جزء معين فى الـ Cycle, فمثلاً أذا كان البرنامج متكون من أربع أفرع فهذا يعنى أن أثناء الـ Cycle يتم قراءة الأربع أفرع بالكامل ولكن إذا تم وضع أمر END فى الفرع الثالث فهذا يعنى أن أثناء الـ Cycle يتم قراءة الثلاث أفرع الأولى فقط أما قراءة ما يوجد فى الفرع الرابع سوف يبقى مرهون بأمر END فإذا كان لا يعمل فسيكون البرنامج طبيعى جداً كما أن أمر END لم يرسم من الأساس إما إذا كان يعمل فسيبقى حاله البرنامج المرسوم فى الفرع الرابع على ما هو علية إلى أن يلغى أمر END مرة أخرى وتتم قراءة الفرع الرابع مرة أخرى.

تمرين عملى باستخدام أمر END:

مكينة تعمل من مكان واحد وتوجد لمبة توضيء فوق المكينة بشرط أن تعمل هذه اللمبة فقط فى الليل.

عدد الدخل	نوع الدخل	أسم الدخل
١	n.c.	I0.0/S1
٢	n.o.	I0.1/S2
٣	n.o.	I0.4/S3 (خلية ضوئية)
عدد التحكمات البرمجية	**نوع التحكمات البرمجية**	**أسم التحكمات البرمجية**
١	END	END
عدد الخرج	**نوع الخرج**	**أسم الخرج**
١	كونتكتور	Q0.1/K1M
٢	اللمبة	Q0.5/K2M

البرنامج:

Network 1 Network Title

Network Comment

I0.1 I0.0 Q0.1

Q0.1

Network 2

I0.4 END

Network 3

Q0.1 Q0.5

ملاحظة:

فى حقيقة الأمر لا يمكن أن نقول بأن أمر END يتحكم فى حجم الدورة **cycle** لأن هذا سيخالف ما قد تم شرحة فى الجزء الأول وهو ما يأكد بأن الـ **cycle time** ثابت لا يتغير, لذالك الذى حدث فى حاله استخدام أمر END لم يكن سوى أن البرنامج قد تظاهر بأنة لم يرى الفرع الرابع لكنة فى جوهر الأمر قد مر على جميع الأفرع بالكامل.

٢- STOP:

كما سبق وذكرنا أن أمر STOP يستخدم لإيقاف وحدة الـ PLC, فأى أن كان عدد الأفرع المتكون منها البرنامج فأنه يتم وضع أمر STOP فى أى فرع من فروع البرنامج فأنه يوقف وحدة الـ PLC, حيث يتم استخدام هذا الأمر فى حالات الطوارئ القسوة التى تستدعى توقف الـ PLC بالكامل, لا يهم كثيراً فى أى فرع سيتم وضع أمر STOP لأن فى جميع الأحوال سوف يتوقف الـ PLC.

تمرين عملى باستخدام أمر STOP:

مكينة تعمل من مكان واحد وفى حالة وجود حريق بالمصنع يجب لوحدة الـ PLC التوقف عن العمل.

عدد الدخل	نوع الدخل	أسم الدخل
١	n.c.	I0.0/S1
٢	n.o.	I0.1/S2
٣	n.o.	I0.7/S3 (حساس حريق)
عدد التحكمات البرمجية	**نوع التحكمات البرمجية**	**أسم التحكمات البرمجية**
١	STOP	STOP
عدد الخرج	**نوع الخرج**	**أسم الخرج**
١	كونتكتور	Q0.1/K1M

البرنامج:

Network 1 Network Title
Network Comment
I0.1 I0.0 Q0.1
Q0.1

Network 2
I0.7 STOP

ملاحظة:

فى حالة استخدام أمر STOP يجب التأكد من إرسال أشارة غير مستمرة لتنفيذ هذا الأمر وذالك لأن فى حاله إرسال إشارة مستمرة على هذا الأمر فهذا يعنى أن كل مرة سوف تقوم فيه بتشغيل وحدة البرمجة سوف تتوقف تلقائياً فلذالك يفضل وضع مفتاح positive edge قبل أمر STOP مباشراً لضمان عدم استمرار الإشارة لفترة طويلة.

٣- DIAG_LED:

كما سبق وذكرنا أن أمر DIAG_LED يستخدم للتحكم بلمبة الـ S.F.,مع ملاحظة أن اللمبة قد تستخدم من قبل وحدة الـ PLC فى حالة أعطال فاضحة كما سوف نشرح بعد قليل وقد تستخدم اللمبة أيضاً من قبل المبرمج كما سوف نشرح فى المثال الحالى.

تضاء اللمبة فقط أذا كانت قيمة الـ IN مختلفة عن صفر, فمثلاً أذا كانت القيمة هى ٧ فستعمل اللمبة.

تمرين عملى باستخدام أمر DIAG_LED:

مكينة تعمل من مكان واحد وفى حالة وجود أى أشارة من الـ Over load فأنة يفصل المكينة ويضيء لمبة الـ .S.F باللون الأصفر.

عدد الدخل	نوع الدخل	أسم الدخل
١	n.c.	I0.0/S1
٢	n.o.	I0.1/S2
٣	n.o.	(Over Load)I0.2/S3
عدد التحكمات البرمجية	**نوع التحكمات البرمجية**	**أسم التحكمات البرمجية**
١	DIAG_LED	DIAG_LED
عدد الخرج	**نوع الخرج**	**أسم الخرج**
١	كونتكتور	Q0.1/K1M

البرنامج:

Network 1 Network Title
Network Comment
I0.1 I0.0 I0.3 Q0.1
Q0.1

Network 2
I0.3 DIAG_LED
EN ENO
7 IN

ملاحظة:

- تضاء اللمبة بالون الأصفر فى حاله استخدامها من قبل المبرمج كما فى التمرين الحالى.
- تضاء اللمبة بالون الأصفر فى حاله تطبيق أمر FORCE على أى عنوان كما سبق وشرحنا فى الفصل الثانى من هذا الكتاب.
- تضاء اللمبة بالون الأحمر فى حاله استخدامها من قبل وحدة الـ PLC فى حالة الأعطال الفاضحة كما سيتم التوضيح فالكتاب التالى الخاص بالأعطال والتمرين العملية.

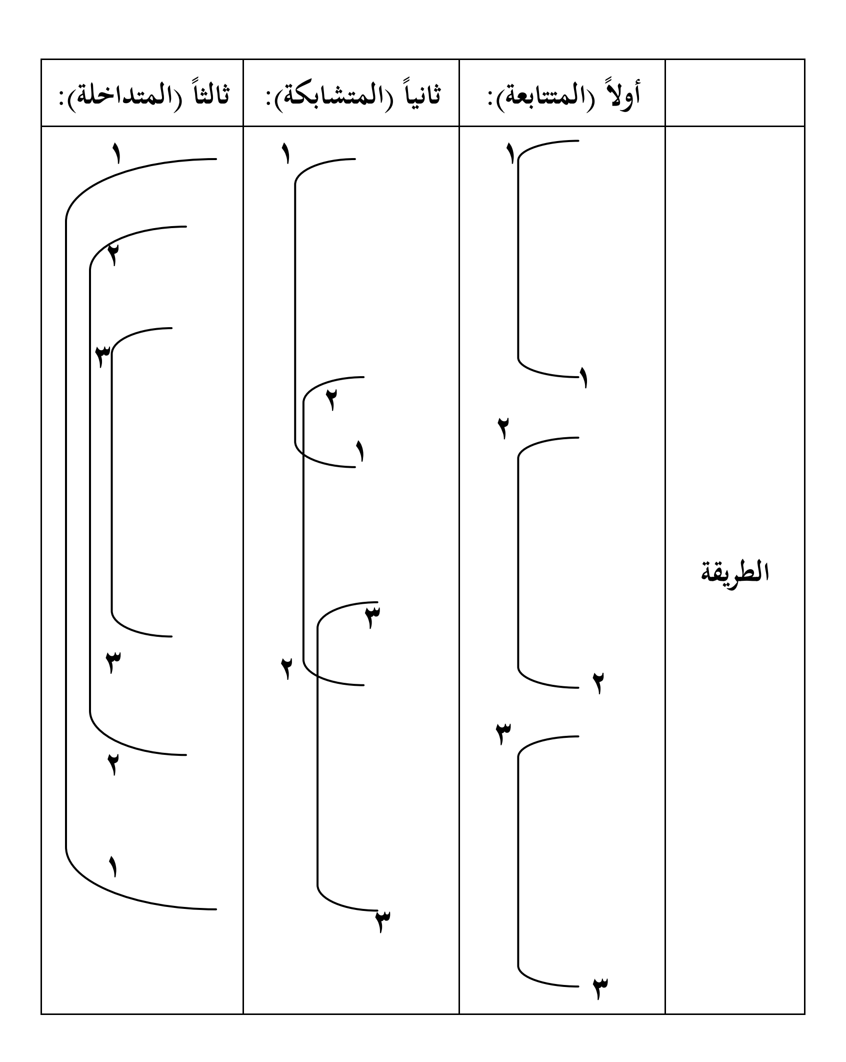
٤- JMP/LBL:

كما سبق وذكرنا أن أمر JMP/LBL يستخدم للتحكم بتنفيذ جزء معين فى وسط الـ Cycle, بحيث ان هذا الأمر يقوم بتنفيذ كل البرنامج ماعدا الأفرع الموجودة بين الـ JMP و الـ LBL مع ملاحظة أن يتم كتابة نفس الرقم على الـ JMP و الـ LBL.

ملاحظة:

- لمعرفة أى JMP يخص أى LBL يكتب نفس الرقم على الأثنان.
- الأرقام المستخدمة تبدأ من صفر إلى ٢٥٥.
- يمكن أن يتكرر الـ JMP و الـ LBL أكثر من مرة ولكن عدد الـ JMP يجب أن يساوى عدد الـ LBL.
- شرط أن يكون الـ JMP فوق الـ LBL وليس العكس.
- يمكن لـ JMP و الـ LBL أن يتكرر بطرق مختلفة: (متتابعة – متشابكة – متداخلة).

الأنواع الثلاثة: المتتابعة – المتشابكة – المتداخلة



	أولاً (المتتابعة):	ثانياً (المتشابكة):	ثالثاً (المتداخلة):
الطريقة			

تمرين عملى باستخدام أمر JMP-LBL:

محرك يعمل فى اتجاهين, ويوجد أيضاً فلاشر يعمل اختيارياً مع المحرك أى أنه يمكن إلغاء الفلاشر.

عدد الدخل	نوع الدخل	أسم الدخل
١	n.c.	I0.0/S1
٢	n.o.	I0.1/S2
٣	n.o.	I1.1/S3
٤	n.c.	I2.2/S4
عدد التحكمات البرمجية	**نوع التحكمات البرمجية**	**أسم التحكمات البرمجية**
١	JMP/LBL	4
عدد الخرج	**نوع الخرج**	**أسم الخرج**
١	كونتكتور	Q0.1/K1M
٢	كونتكتور	Q0.2/K2M
٣	لمبة	Q1.1/K3M
٤	لمبة	Q1.2/K4M

ملاحظة:

- فى حالة استخدام أمر JMP-LBL يتطلب دائماً أن يتم رسم الجزء الخاص بال JMP فوق الجزء الخاص بال LBL وذالك لأنه حين يعمل ال JMP فأنه سيقفز إلى أسفل حتى يصل إلى ال LBL المعنون بنفس عنوان ال JMP.
- فى حالة استخدام أمر JMP-LBL ولكن من دون الألتزام بالشروط السابق ذكرها أى أن فى حالة أن يتم رسم الجزء الخاص بال JMP أسفل الجزء الخاص بال LBL ستحدث مشكلة وذالك لأنه حين يعمل ال JMP فأنه سيقفز إلى الأعلى حتى يصل إلى ال LBL المعنون بنفس رقم ال JMP من ما يستدعى إلى الدوران عكس الحركة الطبيعية للبرنامج فتحدث المشكلة وسوف يتم شرح الأعطال بالتفصيل فى الكتاب التالى الخاص بالأعطال والتمارين العملية.

البرنامج:

Network 1 Network Title

Network Comment

I0.1 I0.0 Q0.2 / Q0.1

Q0.1

Network 2 Network Title

Network Comment

I0.2 I0.0 Q0.1 / Q0.2

Q0.2

Network 3

I1.1 4 JMP

Network 4

Q0.1 T55 <>I 20 I2.2 T55 IN TON

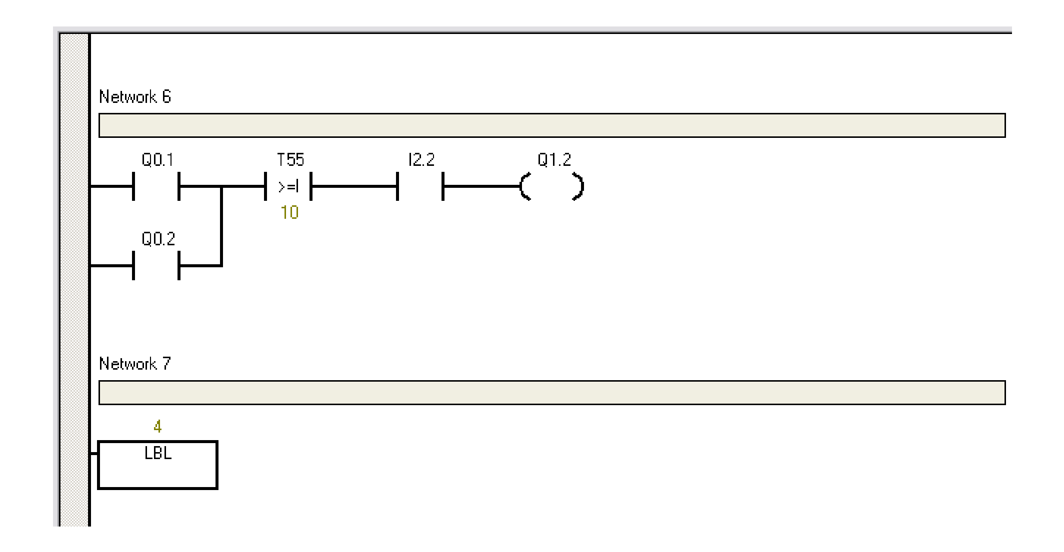
1 PT 100 ms

Q0.2

Network 5

Q0.1 T55 <=I 10 I2.2 Q1.1

Q0.2



٥- RET:

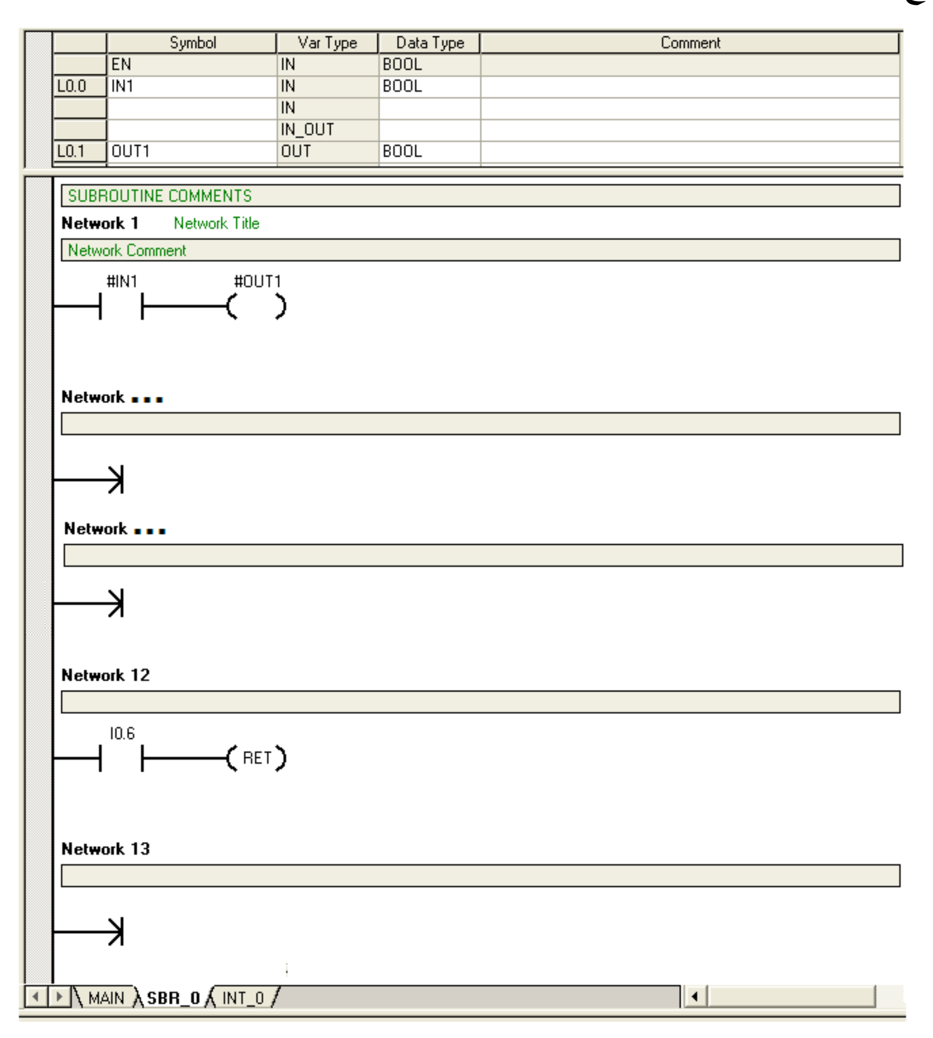
كما سبق وذكرنا أن أمر RET يستخدم للرجوع من صفحة البرمجة الفرعية إلى صفحة البرمجة الرئيسية قبل الأنتهاء من تنفيذ البرنامج الفرعى بالكامل أو حتى بعد الأنتهاء من تنفيذ البرنامج بالكامل.

تمرين عملى باستخدام أمر RET:

تمرين يحتوى على برنامج فرعى وفى حاله الضغط على المفتاح I0.1 فيخرج من صفحة البرنامج الفرعى إلى البرنامج الرئيسى فى الحال دون أن يكمل حتى يصل إلى الفرع الأخير.

عدد الدخل	نوع الدخل	أسم الدخل
١	n.c.	I0.1/S1
عدد التحكمات البرمجية	**نوع التحكمات البرمجية**	**أسم التحكمات البرمجية**
١	RET	RET

البرنامج:



توضيح:

ليس الفكرة فى البرنامج بل الفكرة هى توضيح انه يمكن أن يتم الخروج من البرنامج الفرعى دون الانتظار حتى نهاية الأفرع بالكامل وذالك يحدث فى حالة تشغيل أمر RET فقط.

٦- FOR/NEXT:

كما سبق وذكرنا أن أمر FOR/NEXT يستخدم للتحكم بقراءة جزء معين فى البرنامج أكثر من مرة قبل الأنتهاء من الـ Cycle .

تمرين عملى باستخدام أمر FOR/NEXT:

تمرين يحتوى على برنامج معين وفى حاله الضغط على المفتاح I0.1 فأنه يتم قراءة الفرع الثانى والثالث عشرة مرات متتالية قبل الأنتهاء من الدورة Cycle.

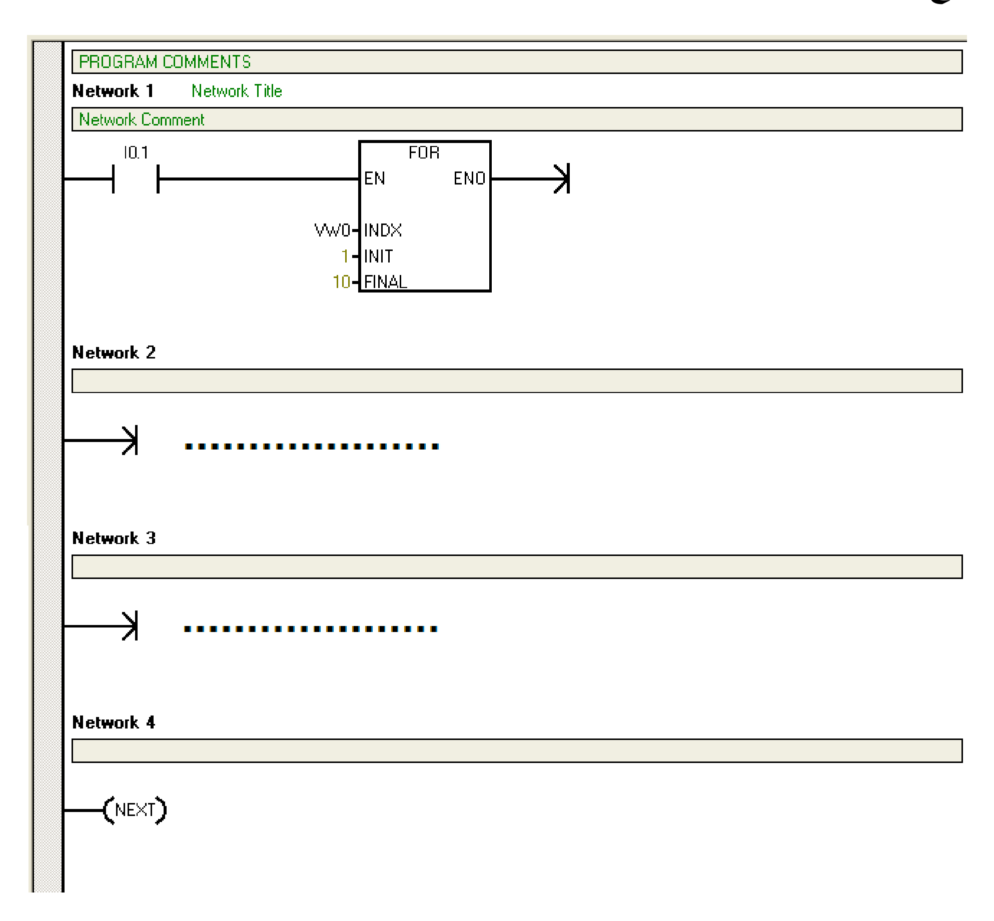
عدد الدخل	نوع الدخل	أسم الدخل
١	n.c.	I0.1/S1
عدد التحكمات البرمجية	**نوع التحكمات البرمجية**	**أسم التحكمات البرمجية**
١	FOR/NEXT	FOR/NEXT

فمثلاً فى التمرين التالى تم تسجيل قيمة ١ فى INIT بينما تم تسجيل قيمة ١٠ فى FINAL فا فى هذه الحالة سيتم المرور على الأفرع المتواجدة بين FOR و NEXT عشرة مرات بحيث أن قيمة INDX سوف يبدأ من واحد إلى عشرة

ملاحظة:

نظراً لأن قيمة INDX سوف تتغير عدة مرات حيث سيضاف ١ إلى القيمة كل دورة فلهذا لا يمكن تحديد أى قيمة مسبقة فى هذا المكان بل سيتم كتابة متغير بحجم word حتى يتيح إلى المحتوى أن يتغير بسهوله.
عند المرور على أمر FOR للمرة الأولى تكون قيمة الـ VW0 ← [١] ثم فى المرة الثانى يضاف إلى الرقم واحد فتصبح قيمة الـ VW0 ← [٢] وهكذا[٣], [٤], [٥], [٦], [٧], [٨], [٩], [١٠]

البرنامج:



أى أن كان البرنامج المرسوم فى الفرع الثانى والثالث فأنه فى حاله غلق المفتاح I0.1 سوف يتم قراءة الأفرعين الثانى والثالث عشرة مرات قبل الانتهاء من الدورة الواحدة.

الباب الحادى العاشر

المحولات

- شرح المحــــــــــــــــــــــــــــــــولات.
- أنواع المحــــــــــــــــــــــــــــــــولات.
- محــــــــــــــــــــــــــــولات لتغير الحجم.
- محــــــــــــــــــــــــولات لتغير الـ format.
- الفــــــــــــــــــرق بين word و integer.
- الفــــــــــــــــرق بين Dword و Dinteger.
- المفاتيح المستخدمة مع المحــــــــــــــــولات.
- الأخطاء الممكن التعرض لها باستخدام المحـولات.
- تمـــــــــــــــــــــــــارين عملية للتوضيح.

المحولات:

تستخدم المحولات للتحويل بثلاث طرق مختلفة:

- **من حجم إلى حجم مختلف.**

فهذا يتحقق في حالة التحويل من Byte إلى Word مثلاً, حيث يتم تغير الحجم من 8bits إلى 16bits.

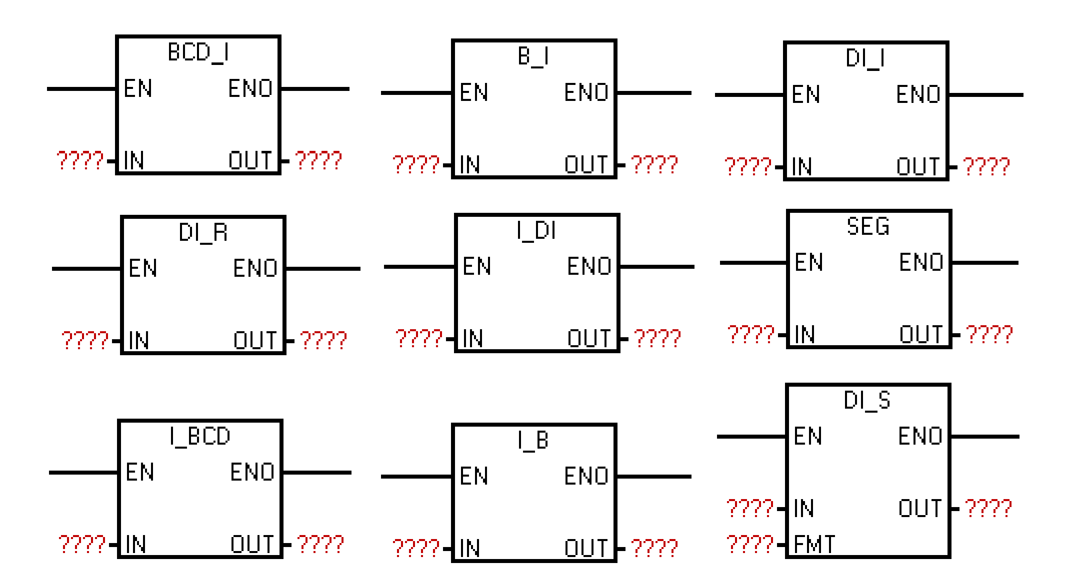
- **من format إلى format أخر.**

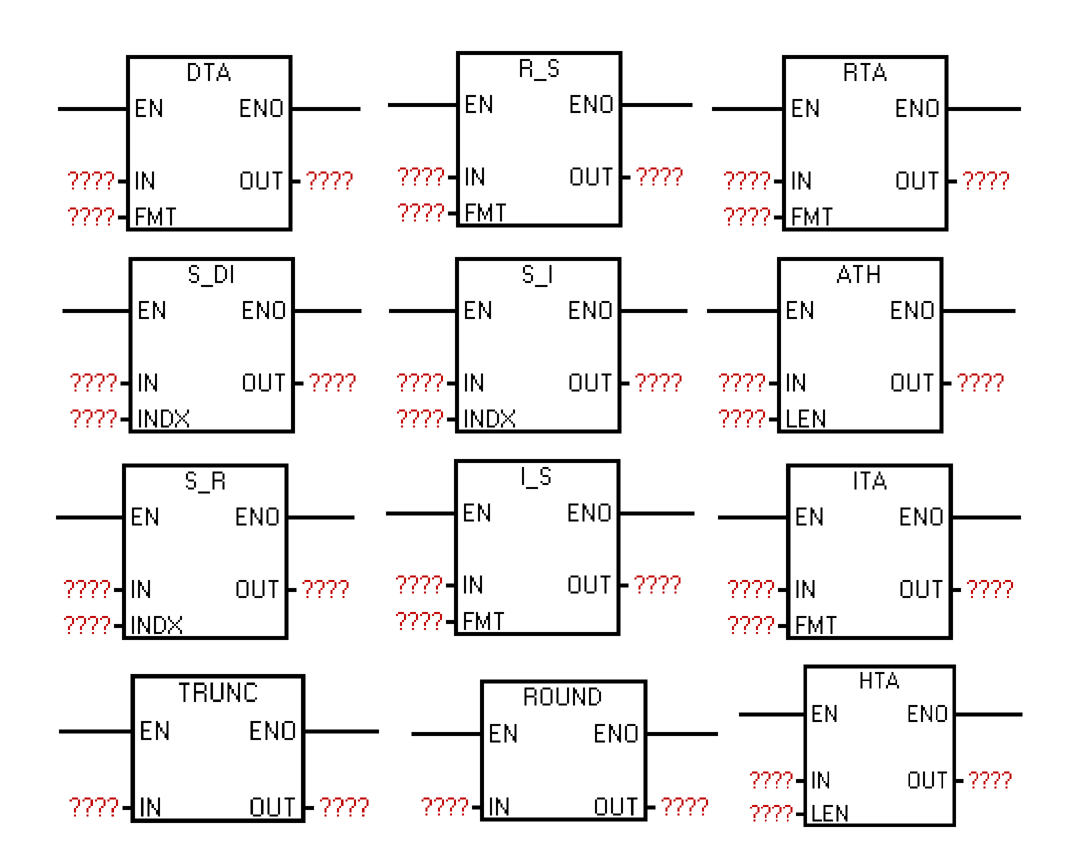
فهذا يتحقق في حالة التحويل من Dinteger إلى Real مثلاً, حيث يتم تغير الـ format من أرقام صحيحة إلى أرقام عشرية دون تغير حجم الذاكرة.

- **من حجم و format إلى حجم و format أخر.**

فهذا يتحقق في حالة التحويل من Byte إلى Integer مثلاً حيث يتم تغير الحجم من 8bits بدون إشارة إلى 16bits بإشارة.

التحويلات المستخدمة في البرمجة:





شرح التحويلات المستخدمة فى البرنامج.

م	الأسم	الشرح	الشكل
١	B_I	يستخدم الـ B_I للتحويل من حجم Byte إلى حجم Word مع الحفاظ على نفس القيمة.	B_I EN ENO ????-IN OUT-????
٢	I_B	يستخدم الـ I_B للتحويل من حجم Word إلى حجم Byte بشرط أن لا تكون القيمة أكبر من أن تكتب على Byte .	I_B EN ENO ????-IN OUT-????

٣	I_DI	يستخدم الـ I_DI للتحويل من حجم Word إلى حجم Dword مع الحفاظ على نفس القيمة.	I_DI EN ENO ????-IN OUT-????
٤	I_S	يستخدم الـ I_S للتحويل من حجم Word إلى حجم Byte بحيث يقوم بتحويل الأرقام إلى أحرف.	I_S EN ENO ????-IN OUT-???? ????-FMT
٥	DI_I	يستخدم الـ DI_I للتحويل من حجم Dword إلى حجم Word بشرط أن لا تكون القيمة أكبر من أن تكتب على Word.	DI_I EN ENO ????-IN OUT-????
٦	DI_R	يستخدم الـ DI_R للتحويل من حجم Dword إلى حجم Dword ولكن يتم تحويل القيمة من رقم صحيح إلى رقم عشرى .	DI_R EN ENO ????-IN OUT-????
٧	DI_S	يستخدم الـ DI_S للتحويل من حجم Dword إلى حجم Byte بحيث يقوم بتحويل الأرقام إلى أحرف.	DI_S EN ENO ????-IN OUT-???? ????-FMT
٨	BCD_I	يستخدم الـ BCD_I للتحويل من حجم Word إلى حجم Word ولكن مع تغير النظام من BCD إلى Integer.	BCD_I EN ENO ????-IN OUT-????

٩	I_BCD	يستخدم الـ I_ BCD للتحويل من حجم Word إلى حجم Word ولكن مع تغير النظام من Integer إلى BCD.	I_BCD: EN ENO, ????-IN OUT-????
١٠	ROUND	يستخدم الـ TRUNC للتحويل رقم عشرى بحجم Dword إلى رقم صحيح بحجم Dword وذالك بالتقريب إلى أقرب رقم صحيح.	ROUND: EN ENO, ????-IN OUT-????
١١	TRUNC	يستخدم الـ TRUNC للتحويل رقم عشرى بحجم Dword إلى رقم صحيح بحجم Dword ولذالك يتم مسح الرقم المكتوب بعد العلامة العشرية.	TRUNC: EN ENO, ????-IN OUT-????
١٢	R_S	يستخدم الـ R_S للتحويل من أرقام عشرية بحجم Dword إلى أحرف بحجم Byte.	R_S: EN ENO, ????-IN OUT-????, ????-FMT
١٣	ITA	يستخدم الـ ITA لتحويل من أرقام صحيحة بحجم Word إلى ما يعادلها في جدول ASCII بحجم Byte.	ITA: EN ENO, ????-IN OUT-????, ????-FMT
١٤	DTA	يستخدم الـ DTA لتحويل من أرقام صحيحة بحجم Dword إلى ما يعادلها في جدول ASCII بحجم Byte.	DTA: EN ENO, ????-IN OUT-????, ????-FMT

١٥	RTA	يستخدم الـ RTA للتحويل من أرقام عشرية بحجم Dword إلى ما يعادلها في جدول ASCII بحجم Byte.	RTA EN ENO ????-IN OUT-???? ????-FMT
١٦	ATH	يستخدم الـ ATH للتحويل من (أرقام, رموز أو أحرف في جدول ASCII) بحجم Byte إلى ما يعادلها في لغة Hexadecimal بحجم Byte أيضاً.	ATH EN ENO ????-IN OUT-???? ????-LEN
١٧	HTA	يستخدم الـ HTA للتحويل من أرقام بلغة Hexadecimal و بحجم Byte إلى ما يعادلها من (أرقام, رموز أو أحرف في جدول ASCII) بحجم Byte أيضاً.	HTA EN ENO ????-IN OUT-???? ????-LEN
١٨	S_I	يستخدم الـ S_I للتحويل من حجم Byte إلى حجم Word بحيث يقوم بتحويل الأحرف إلى أرقام صحيحة.	S_I EN ENO ????-IN OUT-???? ????-INDX
١٩	S_DI	يستخدم الـ S_DI للتحويل من حجم Byte إلى حجم Dword بحيث يقوم بتحويل الأحرف إلى أرقام صحيحة.	S_DI EN ENO ????-IN OUT-???? ????-INDX
٢٠	S_R	يستخدم الـ S_R للتحويل من حجم Byte إلى حجم Dword بحيث يقوم بتحويل الأحرف إلى أرقام عشرية.	S_R EN ENO ????-IN OUT-???? ????-INDX

٢١	SEG	يستخدم الـ SEG لإضاءة اللمبات الخاصة بالـ (SSD) SEVEN SEGMEND DISPLAY حسب الأرقام المكتوبة فى IN التى هى بحجم Byte.	SEG: EN ENO, ????-IN OUT-????

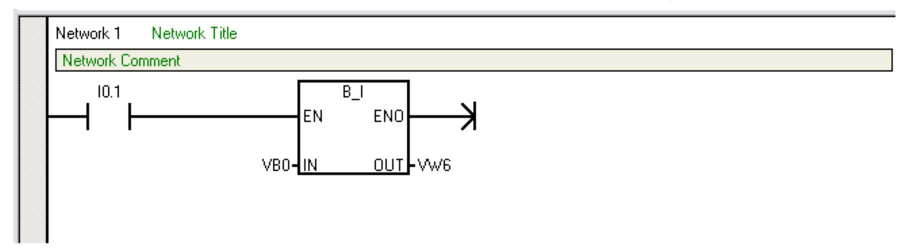
ملاحظة:

كيفية معرفة أذا كان الرقم الذى سيكتب على الذاكرة هو رقم صحيح أم رقم عشرى و كيفية معرفة النظام الذى سيستخدم لعرض القيمة.

م	الأسم	المقصود	الشرح
١	W	WORD	هـذه الـذاكرة مكونـة مـن ١٦ bits ويكتـب عليهـا أرقـام صـحيحة دون إشارة.
٢	I	INTEGER	هـذه الـذاكرة مكونـة مـن ١٦ bits ويكتـب عليهـا أرقـام صـحيحة بإشارة موجبة أو سالبة.
٣	DW	DWORD	هـذه الـذاكرة مكونـة مـن ٣٢ bits ويكتـب عليهـا أرقـام صـحيحة دون إشارة.
٤	DI	DINTIGER	هـذه الـذاكرة مكونـة مـن ٣٢ bits ويكتـب عليهـا أرقـام صـحيحة بإشارة موجبة أو سالبة.
٥	R	REAL	هـذه الـذاكرة مكونـة مـن ٣٢ bits ويكتـب عليهـا أرقـام عشـرية بإشارة موجبة أو سالبة.
٦	S	STRING	هـذه الـذاكرة مكونـة مـن ٨ bits ويكتـب عليهـا أرقـام صـحيحة لتمثيل حرف واحد.

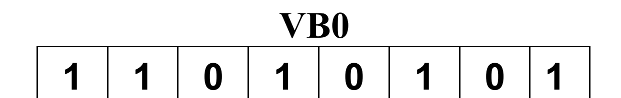
مثال عملي:

مثال عملي باستخدام B_I:



توضيح للشرح:

VB0



1	1	0	1	0	1	0	1

VW6

0	0	0	0	0	0	0	0	1	1	0	1	0	1	0	1

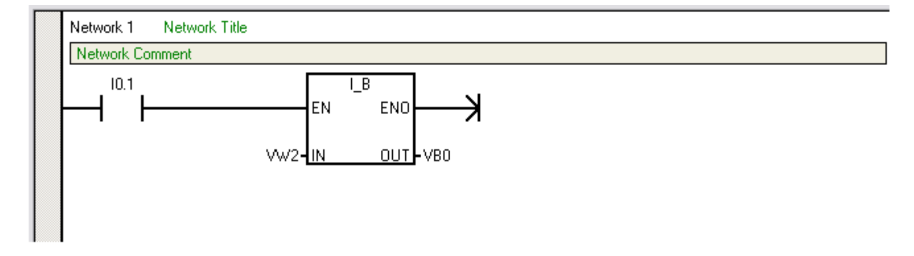
لم تتغير القيمة ولكن تم تغير الحجم من Byte إلى Word.

ملاحظة:

كل ما يكتب على byte يمكن أن ينقل على word لأننا ننقل من الحجم الأصغر إلى الحجم الأكبر ولكن العكس ليس بمضمون لأنه يعتمد على القيمة الفعلية المسجلة داخل الذاكرة.

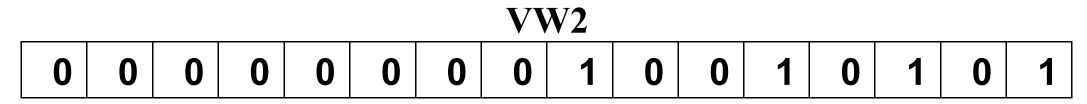
مثال عملي:

مثال عملي باستخدام I_B:



توضيح للشرح:

VW2



0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	1

VB0

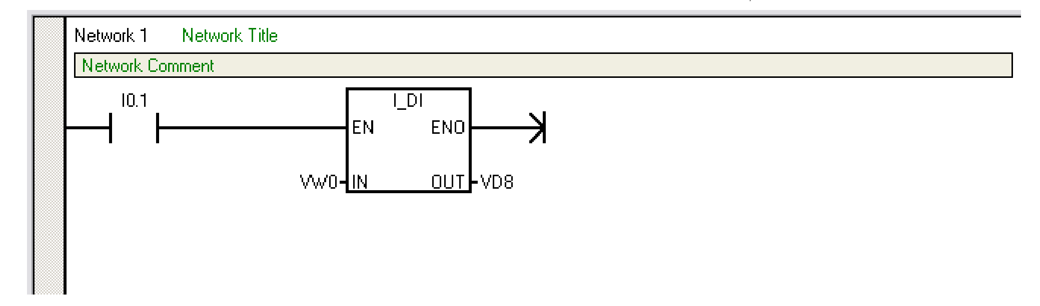
1	0	0	1	0	1	0	1

لم تتغير القيمة ولكن تم تغير الحجم من Word إلى Byte.

يجب مراعاة أن لا يكون محتوى الـ word أكبر من أن يكتب على byte.

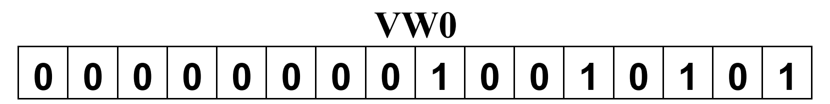
مثال عملي:

مثال عملي باستخدام I_D:



توضيح للشرح:

VW0



0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	1

VD8

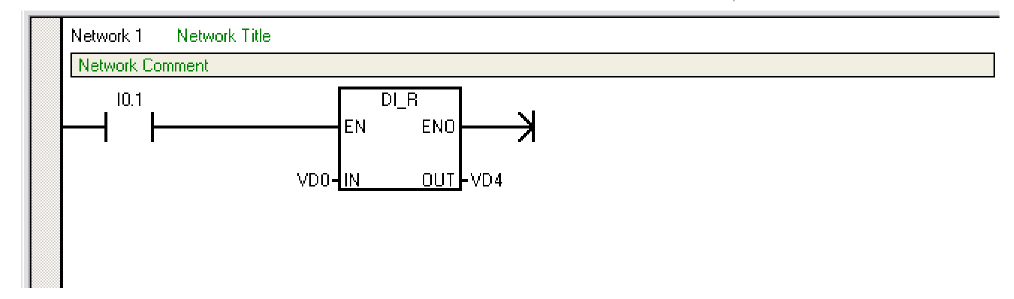
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	1

لم تتغير القيمة ولكن تم تغير الحجم من Word إلى Dword.

كل ما يكتب على Word يمكن أن ينقل على Dword لأننا ننقل من الحجم الأصغر إلى الحجم الأكبر.

مثال عملي:

مثال عملي باستخدام DI_R:



توضيح للشرح:

VD0

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

VD4

0	1	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

لم تتغير القيمة ولم يتغير الحجم ولكن تم تغير النظام من أرقام صحيحة لأرقام عشرية.

مثال عملي:

مثال عملي باستخدام ROUND:

Network 1 Network Title
Network Comment
I0.1 ROUND EN ENO
VD0 IN OUT VD4

توضيح للشرح:

VD0

0	1	0	0	0	0	0	0	1	0	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1

VD4

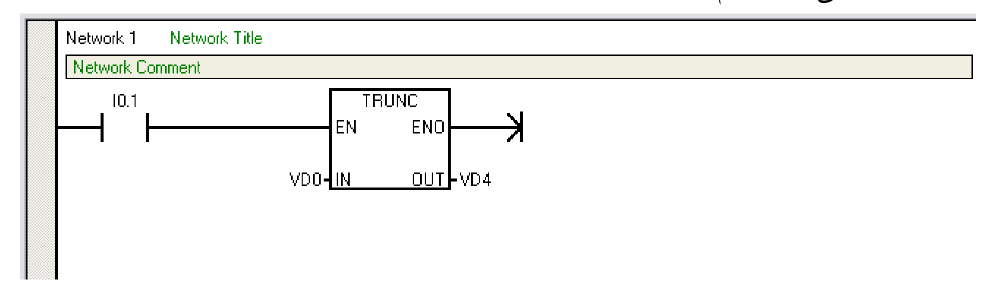
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

أمر ROUND يقرب إلى أقرب رقم صحيح.

الرقم المكتوب فى VD0 هو ٦,٤ بينما المكتوب فى VD4 هو ٦

مثال عملى:

مثال عملى باستخدام TRUNC:



توضيح للشرح:

VD0

0	1	0	0	0	0	0	0	1	0	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1

VD4

0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

أمر TRUNC يمسح الرقم المكتوب بعد العلامة و يكتب الرقم الصحيح فقط.

الرقم المكتوب فى VD0 هو ٦,٤ بينما المكتوب فى VD4 ٤

ما الفائدة من استخدام المحولات؟

١– فى حالة استخدام قيمة من على IB0 للاستخدام مع مؤقت زمنى أو مع عداد فسوف نلاحظ أنه توجد مشكلة وهى أن القيمة الخاصة بالمفاتيح هى على حجم byte بينما المؤقت الزمنى أو العداد يعمل كل منهم على ذاكرة بحجم word ولذالك فأنة يتم استخدام المحولات من byte إلى word وذالك عن طريق العملية B-I.

٢– فى حالة استخدام قيمة من على مؤقت زمنى أو من على عداد للاستخدام مع قيمة أخرى (معادلة رياضية) تحتوى على أرقام عشرية فسوف نلاحظ أنه توجد مشكلة وهى أن القيمة الخاصة بالمؤقت الزمنى أو بالعداد هى بحجم word بينما الأرقام العشرية أى الأرقام غير الصحيحة تكتب على ذاكرة بحجم Dword ولذالك فأنة يتم استخدام:

- **أولاً**: المحولات من word إلى Dword وذالك عن طريق العملية I-DI لتغيير الحجم مع الاحتفاظ بنفس القيمة .
- **ثانياً**: المحولات من Dword إلى Real وذالك عن طريق العملية DI-R لتغيير القيمة الصحيحة لقيمة عشرية دون تغيير الحجم.

٣– فى حالة استخدام قيمة عشرية بحجم Dword فمن المؤكد أن القيمة لن تكون صحيحة بل ستحتوى على أى رقم بعد العلامة العشرية حتى وأن كان هذا الرقم هو صفر وبينما فى حالة التعامل مع عداد ستكون المشكلة ليست فى الحجم فقط بل فى الـ format أيضاً فلذالك سيستخدم:

- **أولاً**: أمر TRUNC لحذف أى أرقام بعد العلامة العشرية دون التغير فى الحجم.
- **ثانياً**: المحولات من Dword إلى Word وذالك عن طريق العملية DI-I لتغيير الحجم دون تغير القيمة.

الباب الثانى عشر

الترحيل و الدوران

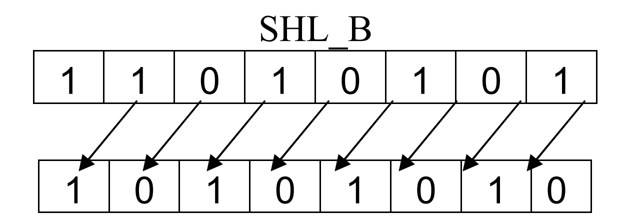
- شرح الترحيـــــــــل SHIFT.
- الأحجام المختلفة للترحيـــــــــل.
- شرح الـــــدوران ROTATE.
- الأحجام المختلفة للـــــــــدوران.
- الفرق بين الترحيـــــل يميناً ويساراً.
- الفرق بين الـــــدوران يميناً ويساراً.
- الأخـــــطاء الممكن التعرض لها.
- تمـــــــــارين عملية للتوضيح.

الترحيل و الدوران:

الترحيل:

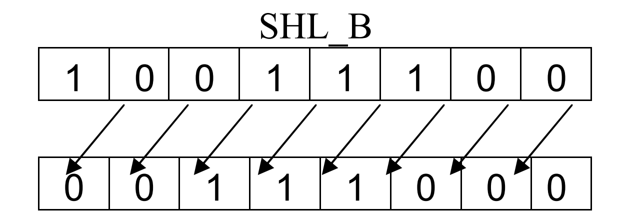
يقوم بترحيل محتويات الـ bits لليمين أو لليسار حسب النوع, بحيث أن محتوى الـ bits التي تخرج عن حدود الذاكرة تمسح تلقائياً.

مثال:



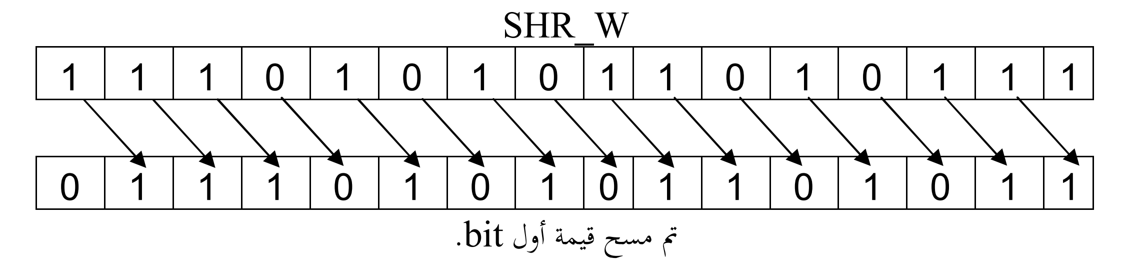
حيث تم مسح قيمة أخر bit.

مثال أخر:



حيث تم مسح قيمة أخر bit.

مثال:



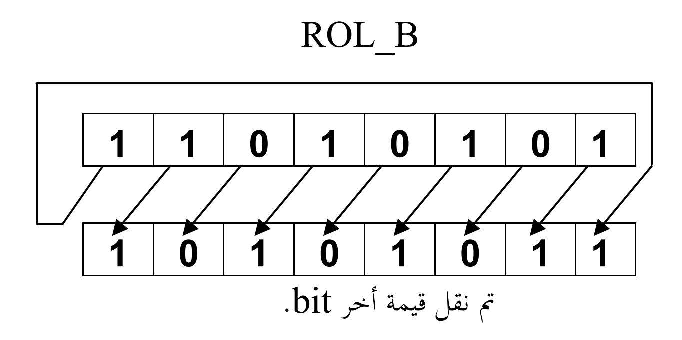
تم مسح قيمة أول bit.

الدوران:

يقوم بدوران محتويات الـ bits لليمين أو لليسار حسب النوع, ولكن محتوى الـ bits التى تخرج من الحجم الذى نتعامل معه لا تمسح بل تعود لنفس الذاكرة من الناحية الأخرى.

مثال:

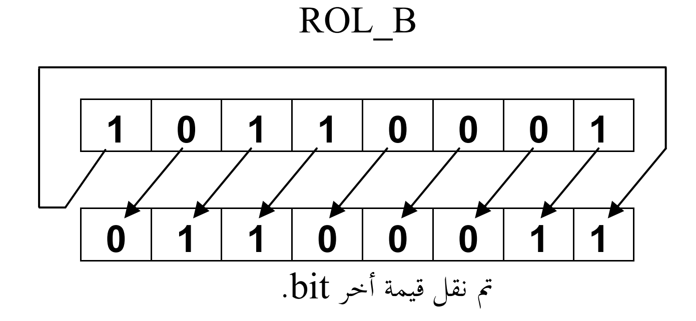
ROL_B



تم نقل قيمة أخر bit.

مثال أخر:

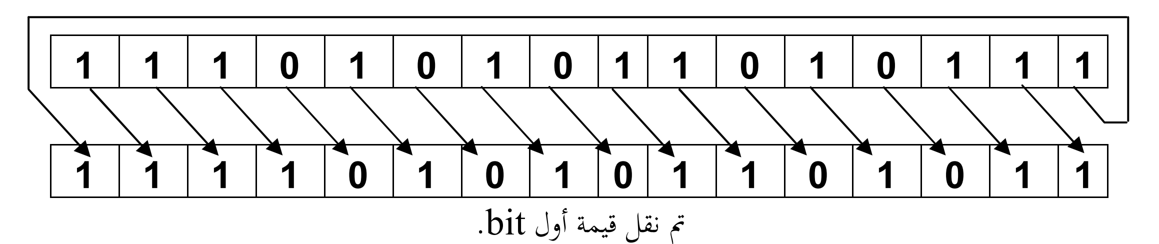
ROL_B



تم نقل قيمة أخر bit.

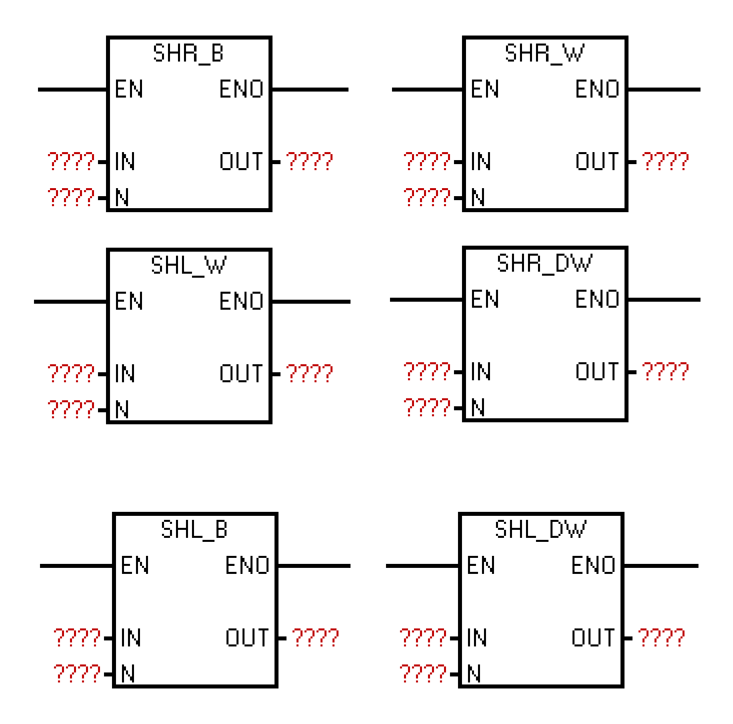
مثال:

ROR_W

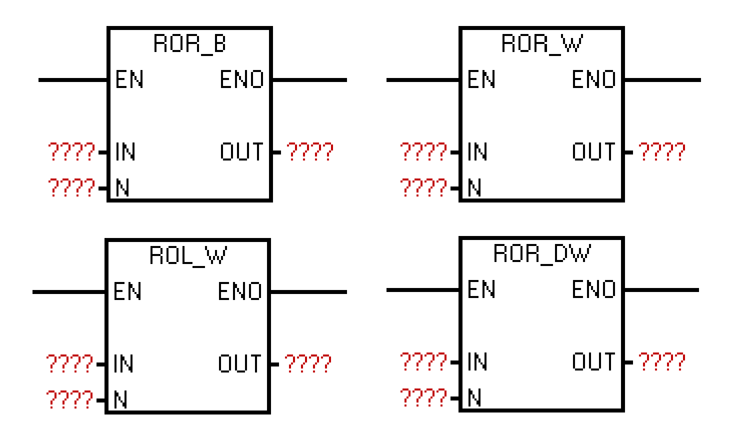


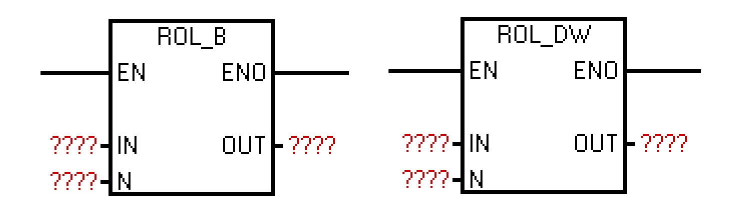
تم نقل قيمة أول bit.

الترحيل:



الدوران





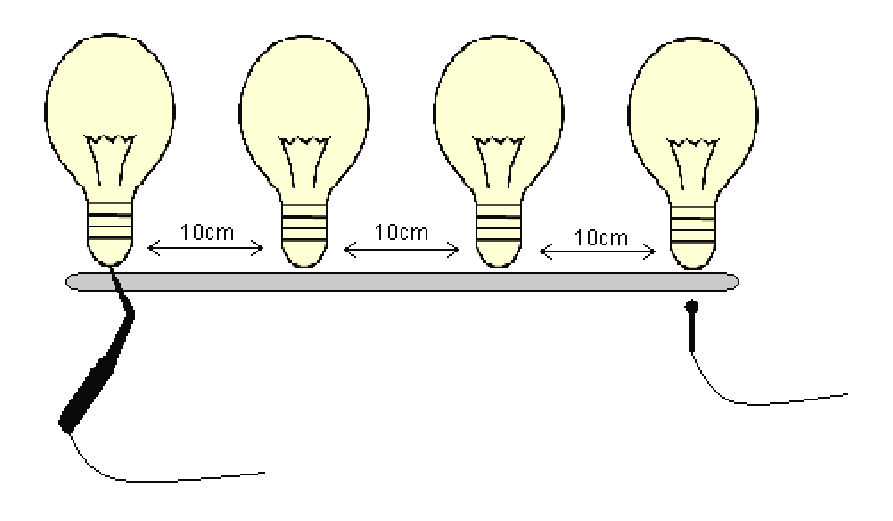
شرح أنواع الترحيل والدوران

م	الأسم	الشرح	الشكل
١	SHR_B	يقوم SHR_B بترحيل القيم الخاصة بال BITS داخل ال BYTE من اليسار إلى اليمين .	SHR_B EN ENO ???? IN OUT ???? ???? N
٢	SHR_W	يقوم SHR_W بترحيل القيم الخاصة بال BITS داخل ال WORD من اليسار إلى اليمين .	SHR_W EN ENO ???? IN OUT ???? ???? N
٣	SHR_DW	يقوم SHR_DW بترحيل القيم الخاصة بال BITS داخل ال DWORD من اليسار إلى اليمين .	SHR_DW EN ENO ???? IN OUT ???? ???? N
٤	SHL_B	يقوم SHL_B بترحيل القيم الخاصة بال BITS داخل ال BYTE من اليمين إلى اليسار.	SHL_B EN ENO ???? IN OUT ???? ???? N

٥	SHL_W	يقوم SHL_W بترحيل القيم الخاصة بال BITS داخل ال WORD من اليمين إلى اليسار.	SHL_W EN ENO ????-IN OUT-???? ????-N
٦	SHL_DW	يقوم SHL_DW بترحيل القيم الخاصة بال BITS داخل ال DWORD من اليمين إلى اليسار.	SHL_DW EN ENO ????-IN OUT-???? ????-N
٧	ROR_B	يقوم ROR_B بدوران القيم الخاصة بال BITS داخل ال BYTE من اليسار إلى اليمين .	ROR_B EN ENO ????-IN OUT-???? ????-N
٨	ROR_W	يقوم ROR_W بدوران القيم الخاصة بال BITS داخل ال WORD من اليسار إلى اليمين .	ROR_W EN ENO ????-IN OUT-???? ????-N
٩	ROR_DW	يقوم ROR_DW بدوران القيم الخاصة بال BITS داخل ال DWORD من اليمين إلى اليسار.	ROR_DW EN ENO ????-IN OUT-???? ????-N

م	التعليمة	الوصف	الرمز
١٠	ROL_B	يقوم ROL_B بدوران القيم الخاصة بال BITS داخل ال BYTE من اليمين إلى اليسار.	ROL_B: EN ENO / ????-IN OUT-???? / ????-N
١١	ROL_W	يقوم ROL_W بدوران القيم الخاصة بال BITS داخل ال WORD من اليمين إلى اليسار.	ROL_W: EN ENO / ????-IN OUT-???? / ????-N
١٢	ROL_DW	يقوم ROL_DW بدوران القيم الخاصة بال BITS داخل ال DWORD من اليمين إلى اليسار.	ROL_DW: EN ENO / ????-IN OUT-???? / ????-N

رسم توضيحي للتمرين العملي:

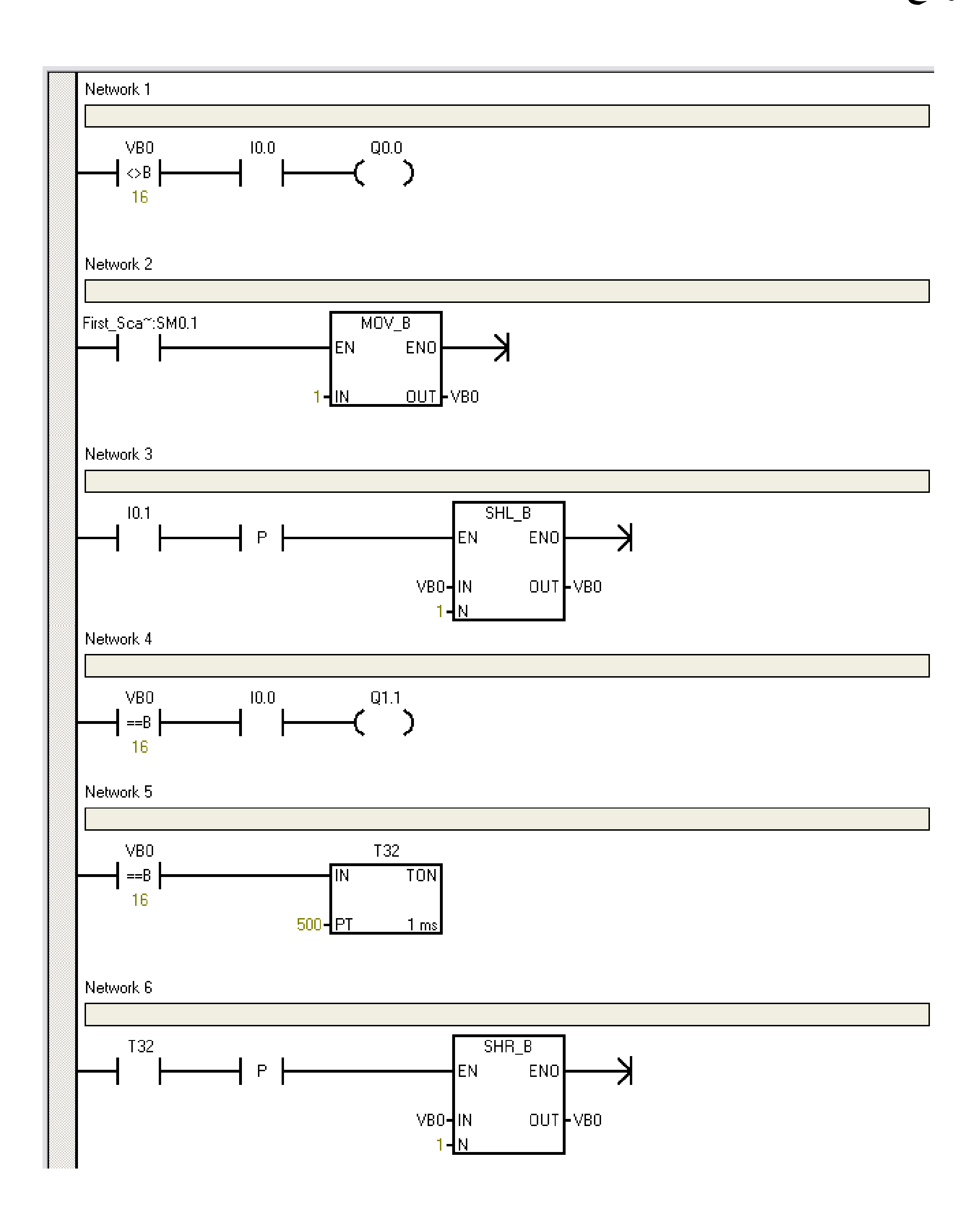


التمرين عملى:

مكينة تقوم بلحام الأطراف الداخلية المثبتة فى القعدة الخاصة باللمبة و لكن نظراً لشدة درجة الحرارة بالقرب من مكينة اللحام سوف يتم و ضع الحساس بعيداً عن مكان مكينة اللحام, المسافة بين اللمبات متساوية وهى تعتمد على حركة السير فهو يتحرك نصف ثانية ويقف نصف ثانية.

عدد الدخل	نوع الدخل	أسم الدخل
١	n.c.	I0.0/S1
٢	n.o.	I0.1/S2
عدد المؤقتات الزمنية	**نوع المؤقتات الزمنية**	**أسم المؤقتات الزمنية**
١	TON	T32
عدد المتغيرات	**نوع المتغيرات**	**أسم المتغيرات**
١	Byte	VB0
عدد الترحيل	**نوع الترحيل**	**أسم الترحيل**
١	SHL_B	VB0
٢	SHR_B	VB0
عدد مفاتيح المقارنة	**نوع مفاتيح المقارنة**	**أسم مفاتيح المقارنة**
١	==B	VB0
عدد الخرج	**نوع الخرج**	**أسم الخرج**
١	كونتكتور	Q0.0/K1M
٢	كونتكتور	Q1.1/K2M

البرنامج:


Network 1
VB0
<>B
16
I0.0
Q0.0
Network 2
First_Sca~:SM0.1
MOV_B
EN
ENO
1
IN
OUT
VB0
Network 3
I0.1
P
SHL_B
EN
ENO
VB0
IN
OUT
VB0
1
N
Network 4
VB0
==B
16
I0.0
Q1.1
Network 5
VB0
==B
16
T32
IN
TON
500
PT
1 ms
Network 6
T32
P
SHR_B
EN
ENO
VB0
IN
OUT
VB0
1
N

الشرح:

:Network1

سوف يعمل السير طالما قيمة الـ VB0 مختلفة عن ١٦.

:Network2

فى أول Cycle سوف تكون قيمة VB0 هى ١ أى أن قيمة V0.0 هي ١.

:Network3

كل مرة تمر اللمبة أمام الحساس سوف يتم تطبيق مبدأ ترحيل الـ bits الخاصة بالـ Byte VB0 لليسار.

:Network4

عندما تصبح قيمة الـ VB0 هى ١٦ هذا يعنى أنه توجد الآن لمبة أمام مكينة اللحام.

:Network5

سوف يعمل المؤقت الزمنى لكى يقوم بفصل مكينة اللحام تلقائياً.

:Network6

فيقوم المؤقت الزمنى بترحيل الـ bits الخاصة بالـ Byte VB0 لليمين فيعمل السير الذى يحمل اللمبات مرة أخرى, وهكذا.

الباب الثالث عشر

العلامات

- شرح العــــــــــــــــــلامات.
- الهدف من استخدام العـــــلامات.
- كيفية التنقل بين الأفــــــرع بسرعة.
- المفاتيح المستخدمة مع العــلامات.
- تمـــــــــــــارين عملية للتوضيح.

العلامات:

تستخدم العلامات لتميز الأفرع عن بعضها و للتنقل من فرع إلى أخر بسهولة خاصة فى البرامج الكبيرة عندما تكون هناك فروع برمجة مرتبطة ببعضها و فى نفس الوقت تبعد عن بعضها من حيث تواجدها فى البرنامج, قد يحدث فى أى برنامج إلى أن نضطر إلى تغير ترتيب أماكن أفرع البرمجة حتى يعمل البرنامج بصورة صحيحة ولكن حين يتطلب الموضوع إلى العودة فى المستقبل للعمل على نفس البرنامج مرة أخرى نكون لم نعد نتذكر بعد أى من أفرع البرمجة مرتبطة ببعضها وهذا يتطلب وقت كبير لإعادة قراءة البرنامج بالكامل من البداية, الأمر الذى كان يمكن حله من البداية فى حاله تم استخدام العلامات كما سنوضح الآن.

المفاتيح المستخدمة:

١- مفتاح وضع العلامات Toggle bookmark

حيث يتم الوقوف على الفرع المراد ثم الضغط على المفتاح الخاص بوضع العلامات.

٢- مفتاح للتنقل للعلامات التالية Next bookmark

حيث يتم التنقل للإمام مباشراً إلى الفرع المحدد دون ضرورة المرور على باقى الأفرع.

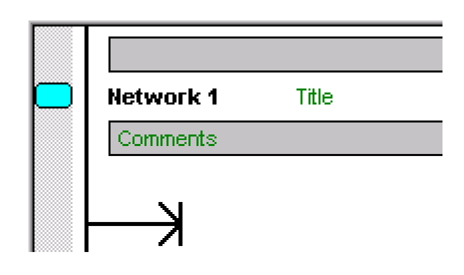
٣- مفتاح للتنقل للعلامات السابقة Previous bookmark

حيث يتم التنقل للخلف مباشراً إلى الفرع المحدد دون ضرورة المرور على باقى الأفرع.

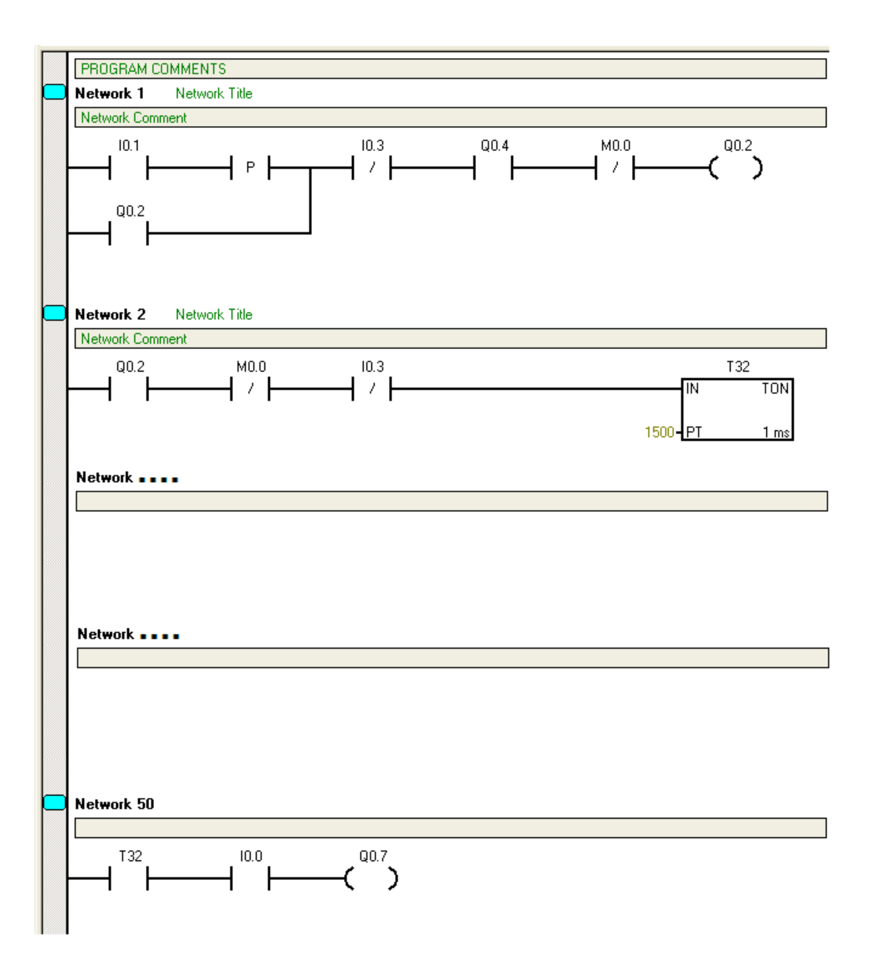
٤- مفتاح حذف جميع العلامات Remove all bookmarks

حيث يتم حذف جميع العلامات من جميع الأفرع الموجودة بالبرنامج.

الشكل العام للعلامات, لل Bookmarks:



البرنامج:



توضيح:

ليس الفكرة فى البرنامج بل الفكرة انه يمكن أن يتم التنقل من الفرع الثانى إلى الفرع الخمسين فى الحال, فبالرغم من أن الفرع الخمسين يحتوى على الخرج الذى يعمل بواسطة المؤقت الزمنى المتواجد فى الفرع الثانى ولكن لظروف ذات علاقة بالأولويات لم نتمكن من وضعهما بالقرب من بعضهما ولكن تم حل المشكلة باستخدام العلامات Bookmarks.

يتم وضع هذه العلامة بترتيب عند الأفرع المرتبطة ببعضها.

الفهرس

الفهرس

الفهرس

الفهرس

الفهرس

الكتب التي صدرت عن معهد السالزيان الإيطالى "دون بوسكو"

الكتاب	المؤلف
🕮 **محركات, مولدات و محولات التيار المتردد**	وجيه جرجس
🕮 **دوائر التحكم الآلي** الجزء الأول	وجيه جرجس
🕮 **دوائر التحكم الآلي** الجزء الثاني	وجيه جرجس
🕮 **الغسالة الفول أوتوماتك** الجزء الأول	وجيه جرجس
🕮 **الغسالة الفول أوتوماتك** الجزء الثاني	وجيه جرجس
🕮 **الدوائر العملية للضغوط الهوائية و الكهروهوائية**	وجيه جرجس
🕮 **غسالة الأطباق**	وجيه جرجس
🕮 **زانوسى الموديلات القديمة 14-16-18بروجرام**	وجيه جرجس
🕮 **موديلات الغسالة كريازى**	وجيه جرجس
🕮 **الدوائر الكهربائية للتركيبات المنزلية**	نبيل رزق
🕮 **صيانة وإصلاح الأجهزة المنزلية**	نبيل رزق
🕮 **أفكار التكيف و التبريد للدوائر الميكانيكية**	إميل فتح الله
🕮 **أفكار التكيف و التبريد للدوائر الكهربائية**	إميل فتح الله
🕮 **أفكار التكيف و التبريد الخدمة والأعطال**	إميل فتح الله
🕮 **برمجة التحكم المنطقي .P.L.C** الجزء الأول	ريمون كمال
🕮 **برمجة التحكم المنطقي .P.L.C** الجزء الثاني	ريمون كمال
🕮 **برمجة التحكم المنطقي .P.L.C** إعطال و تمارين عملية	تحت التحضير